**TUGAS AKHIR - TE 141599**

# PERANCANGAN KONTROLER MPC UNTUK TRAJECTORY TRACKING PADA GERAK *CRUISE QUADCOPTER*

Astrid Rachma Pratiwi
NRP 2212 100 103

Dosen Pembimbing
Ir. Rusdhianto Effendie A.K., MT.
Eka Iskandar, ST., MT.

JURUSAN TEKNIK ELEKTRO
Fakultas Teknologi Industri
Institut Teknologi Sepuluh Nopember
Surabaya 2016


**FINAL PROJECT - TE 141599**

# *MPC CONTROLLER DESIGN FOR TRAJECTORY TRACKING OF CRUISE MOVEMENT QUADCOPTER*

Astrid Rachma Pratiwi
NRP 2212 100 103

*Supervisors*
Ir. Rusdhianto Effendie A.K., MT.
Eka Iskandar, ST., MT.

DEPARTMENT OF ELECTRICAL ENGINEERING
Faculty of Industrial Technology
Sepuluh Nopember Institute of Technology
Surabaya 2016

# PERANCANGAN KONTROLER MPC UNTUK TRAJECTORY TRACKING PADA GERAK CRUISE QUADCOPTER

**TUGAS AKHIR**

**Diajukan Guna Memenuhi Sebagian Persyaratan Untuk**
**Memperoleh Gelar Sarjana Teknik Elektro**
**Pada**
**Bidang Studi Sistem Pengaturan**
**Jurusan Teknik Elektro**
**Institut Teknologi Sepuluh Nopember**

**Menyetujui**

| Dosen Pembimbing I, | Dosen Pembimbing II, |
|---|---|
| **Ir. Rusdhianto Effendie A.K., MT.** | **Eka Iskandar, ST., MT.** |
| **NIP. 195704241985021001** | **NIP. 198005282008121001** |

**SURABAYA**
**JUNI, 2016**

# PERANCANGAN KONTROLER MPC UNTUK *TRAJECTORY TRACKING* PADA GERAK *CRUISE QUADCOPTER*

Astrid Rachma Pratiwi
2212 100 103

Dosen Pembimbing I : Ir. Rusdhianto Effendie A.K., M.T
Dosen Pembimbing II : Eka Iskandar, ST, MT

## ABSTRAK

Salah satu permasalahan kendali pada *quadcopter* adalah *trajectory tracking* dimana *quadcopter* dapat mejejak jalur yang sudah ditetapkan. Pada penelitian Tugas Akhir ini dirancang pengendalian gerak *cruise quadcopter* saat trajectory tracking menggunakan Kontroler PID untuk gerak rotasi dan *Model Predictive Control* (MPC) untuk gerak translasi. Gerakan Cruse *quadcopter* saat trajectory tracking merupakan gerakan manuver yang mencakup perubahan sumbu X, Y dan Z. Referensi jalur yang akan disimulasikan untuk trajectory tracking gerak *cruise quadcopter* berbentuk lingkaran. Penentuan parameter MPC dengan memvariasikan salah satu nilai parameter MPC yaitu *pediction horizon* (Np). Didapatkan nilai parameter MPC Np_x sebesar 20, Np_y sebesar 14, dan Np_z sebesar 10 dengan nilai *control horizon* sebesar Nc_x sebebsar 2, Nc_y sebesar 2, dan Nc_z sebesar 3. Respon tersebut memiliki RMSE pada sumbu X sebesar 3,3%, RMSE pada sumbu Y sebesar 2,3% dan RMSE pada sumbu Z sebesar 1,5% dengan lagging pada sumbu X sebesar 0,48 detik, sumbu Y sebesar 0,3 detik dan pada sumbu Z sebesar 0,32 detik.

**Kata Kunci :** *Quadcopter*, Trajectory Tracking, Gerak *Cruise,* Kontroler *PID, Model Predictive Control*

*[Halaman ini sengaja dikosongkan]*

# *MPC CONTROLLER DESIGN FOR TRAJECTORY TRACKING OF CRUISE MOVEMENT QUADCOPTER*

Astrid Rachma Pratiwi
2212 100 103

*Supervisor I* : Ir. Rusdhianto Effendie A.K., M.T
*Supervisor II* : Eka Iskandar, ST, MT

## *ABSTRACT*

*One of quadcopter control problem issue is the control of the trajectory tracking quadcopter which can track the path that has been set. In this final assignment research trajectory tracking on cruise movement quadcopter using PID controller for rotational motion and Model Predictive Control (MPC) for the translational motion. Cruse movement quadcopter current trajectory tracking is a maneuver which includes a change axes X, Y and Z. Reference track to be simulated for trajectory tracking quadcopter circle. Determining parameters of MPC WITH varying parameter value MPC prayer One That pediction horizon (Np). MPC Np_x obtained parameter value by 20, Np_y at 14, and by 10 WITH Np_z horizon control value of Nc_x is 2, Nc_y by 2, and Nc_z of 3. The response has RMSE on X-axin by 3,3%,RMSE on Y-axis by 2,3% and on Z-axis at 1,5%. with lagging on X-axis of 0,48 and Y of 0.3 seconds and on-axis Z of 0.32 seconds.*

***Keyword*** **:** *Quadcopter*, *Trajectory Tracking*, *Cruise Movement, PID Controller, Model Predictive Control*

*[Halaman ini sengaja dikosongkan]*

# DAFTAR ISI

# DAFTAR GAMBAR

# DAFTAR TABEL

*[Halaman ini sengaja dikosongkan]*

# BAB I
# PENDAHULUAN

## 1.1 Latar Belakang

Saat ini, UAV (*Unmanned Aerial Vehicle*) yang sedang mengalami perkembangan yang pesat dan memiliki potensi yang sangat besar, baik untuk keperluan militer maupun sipil. Banyak sekali jenis UAV, salah satunya adalah *quadcopter*, peawat yang memiliki empat rotor atau empat baling-baling, jenis pesawat ini memiliki keunggulan lebih dibanding helicopter biasa [1]. Helicopter biasa menggunakan 1 rotor penggerak untuk menggerakkan 2 baling-baling sedangkan *quadcopter* mempunyai 4 rotor yang fokus untuk menggerakkan masing-masing propeller, sehingga kinerja rotor *quadcopter* lebih efektif.

*Quadcopter* saat ini menjadi bahan perbincangan dan penelitian bagi para peneliti dikarekan memiliki manuverabilitas yang tinggi, rancangan yang sederhana dan kelengkapan sensor yang digunakan. *Quadcopter* juga sangat efektif dalam melakukan pengawasan, pemeriksaan area, dan pemetaan pada daerah yang tidak bisa terjangkau oleh manusia. Contohnya, pada bidang pertanian, *quadcopter* dapat digunakan untuk meningkatkan manajemen penanaman pada lahan pertanian seperti mengamati dan memantau lahan yang akan di tanam.

*Quadcopter* mempunyai dua macam gerakan, yaitu gerakan rotasi dan translasi. Gerak rotasi merupakan gerakan yang terjadi pada poros *quadcopter* dimana tebrdiri dari gerak *roll*, *pitch* dan *yaw*. Sedangkan gerak translasi *quadcopter* mempunyai dua gerakan, yaitu gerakan lateral dan gerakan longitudinal. Gerakan lateral merupakan gerakan *quadcopter* saat berbelok ke kanan maupun ke kiri menuju arah sumbu x dan y. Sedangkan gerakan longitudinal merupakan gerakan *quadcopter* pada sumbu z, pada saat *quadcopter* akan menuju ke atas atau ke bawah.

Permasalahan kendali pada *quadcopter* salah satunya adalah trajectory tracking dimana *quadcopter* dapat menjejak jalur referensinya. Dimana *quadcopter* melakukan gerak tanslasi maju maupun menyamping untuk menjejak jalur referensi. Gerak translasi pada quaqcopter di tentukan oleh resultan gaya dan sudut-sudut pada *quadcopter*. Jika dilakukan pengendalian gerak translasi, maka sinyal kontrol bergantung pada evaluasi gerak translasinya dan besar sudut-sudut quadrotor [2].

Untuk pengendalian trajectory tracking *quadcopter*, sudah banyak teknik kontrol yang digunakan. Kontrol PID, LQR dan *Model Predictive Control* (MPC) baru-baru ini telah banyak digunakan. Kontroler PID

adalah pilihan yang paling populer untuk mengendalikan semacam proses, tetapi untuk sistem MIMO seperti *quadcopter* sulit untuk menggunakan kontroler ini sedangkan kontroler LQR adalah metode pengendalian yang relatif modern tetapi terbatas pada aplikasi dimana sistem adalah model yang linier. Apabila menggunakan metode ini untuk mengontrol sistem nonlinier maka sistem harus di diubah menjadi bentuk linier terlebih dahulu. Untuk mengevaluasi kinerja kontroler LQR, kita perlu membandingkannya dengan hasil kontroler yang lain yang membutuhkan kerumitan yang lain pula [3]. Di sisi lain , MPC dapat menangani sistem linear dan sistem nonlinier [4]. Dibandingkan dengan PID , Tuning MPC lebih mudah bahkan untuk sistem MIMO yang kompleks. Sehingga pada penelitian ini diharapkan kontroler MPC dapat mengoptimal gerak *cruise quadcopter*.

## 1.2 Perumusan Masalah

Pada tugas akhir ini dimaksudkan untuk menyelesaikan permasalah trajectory tracking atau bisa disebut permasalahan suatu kemampuan *quadcopter* untuk mengikuti lintasan umum yang sudah ada, pada gerak *cruise quadcopter*. Dimana *quadcopter* bergerak maneuver yang dapat menyebabkan posisi sumbu X,Y dan Z berubah, pergerakan tersebut juga merubah posisi sudut *quadcopter*. Sehingga sudut dari *quadcopter* akan dijadikan parameter untuk pergereakan tersebut. Maka untuk pengendalian trcking *quadcopter* dibutuhkan kontroler yang mampu beradaptasi dengan perubahan parameter.

## 1.3 Batasan Masalah

Pada Tugas Akhir ini, parameter dari *quadcopter* yang digunakan adalah *quadcopter* rakitan, yang berada pada Laboratorium Teknik Pengaturan B-105, Jurusan Teknik Elektro ITS. Pada penelitian ini, dilakukan pengendalian gerakan *cruise trajectory tracking* pada *quadcopter*. Akan ada 6 variabel yang akan dikontrol, yaitu posisi *quadcopter* pada sumbu X, posisi *quadcopter* pada sumbu Y, posisi *quadcopter* pada sumbu Z, serta sudut *roll*, *pitch*, dan *yaw* dari *quadcopter*.

Pada pengaturan posisi sudut *quadcopter roll*, *pitch*, dan *yaw* digunakan kontroler *PID* . Sedengkan, untuk pengaturan sudut posisi *quadcopter* pada sumbu X,Y,dan Z digunakan kontroler MPC. Kedua kontroler ini disambungkan secara *cascade*, dengan *outer loop* adalah kontroler posisi, dan *inner loop* adalah kontroler kecepatan.

## 1.4 Tujuan

Tujuan dari pelaksanaan tugas akhir, antara lain :

1. Mengindentifikasi kinematika dan dinamika *quadcopter* sehingga didapatkan model matematika sistem
2. Merancang kontroler *Model Predictive Control* untuk pengendalian posisi dari *quadcopter* dan kontroler PID untuk stabilitas *quadcopter*.
3. Menganalisa respon simulasi *quadcopter* dengan mensimulasikannya sehingga diketahui tingkat keakuratan dari kontroler.

Hasil yang diperoleh dari pelaksanaan tugas akhir diharapkan dapat dijadikan acuan dalam pembuatan sistem kontrol untuk gerak *cruise quadcopter*.

## 1.5 Metodologi

Metodologi yang digunakan dalam tugas akhir ini terdiri dari beberapa tahap sebagai berikut:

a. Studi literatur

Tahap pertama adalah mencari informasi atau data mengenai *plant*, kontroler, atau sistem secara keseluruhan. Studi literatur diperlukan sebagai landasan dalam mengerjakan Tugas Akhir agar diperoleh teori penunjang yang memadai, baik ilmu dasar, analisis, maupun metode penelitian.

b. Identifikasi *plant*

Pada tahap kedua, Identifikasi *plant quadcopter* dilakukan dengan pengukuran *input* - *output*, sehingga diperoleh parameter-parameter yang ada pada *quadcopter* untuk memperoleh model matematikanya. Dengan didapatkannya model matematika dari *quadcopter*, perancangan kontroler dapat dilakukan.

c. Perancangan Kontroler

Tahap ketiga adalah perancangan kontroler untuk pengendalian gerak *quadcopter*. Perancangan kontroler PID untuk pengendalian gerak rotasi sedangkan perancangan kontroler MPC untuk pengendalian gerak translasi.

d. Simulasi dan Analisis Data

Tahap keempat adalah mensimulasikan kontroler yang telah dirancang menggunakan *software* MATLAB 2015a agar mengetahui performansi dari sistem yang telah diberi kontroler. Setelah data diperoleh maka dilakukan analisa data.

e. Penarikan Kesimpulan
Analisis data yang telah didapatkan akan digunakan untuk penarikan kesimpulan.

f. Pembuatan Laporan
Tahap ini merupakan tahap akhir dari serangkaian pelaksanaan tugas akhir. Penyusunan buku tugas akhir dilakukan sebagai bentuk laporan tertulis dari proses dan hasil kerja terkait topik yang diusulkan

## 1.6 Sistematika Pembahasan

Pembahasan dalam tugas akhir ini akan dibagi dalam lima bab dengan sistematika sebagai berikut:

**BAB I PENDAHULUAN**
Pada bab ini, dibahas mengenai latar belakang, perumusan permasalahan, tujuan, metodologi, sistematika penulisan, dan relevansi dari Tugas Akhir.

**BAB II TEORI PENUNJANG**
Bab ini berisi tentang teori-teori yang berkaitan dengan topik yang diangkat dalam Tugas Akhir ini. Teori yang dibahas pada bab ini adalah mengenai dinamika dan kinematika *quadcopter*, Linearisasi, Kontroler PID dan *Model Predictive Control*.

**BAB III PERANCANGAN SISTEM**
Pada bab ini akan dijelaskan mengenai perancangan algoritma kontrol *Trajectory Tracking* dengan *Model Predictive Control* berdasarkan teori pada Bab II.

**BAB IV HASIL DAN PEMBAHASAN**
Bab ini berisi tentang hasil pengujian simulasi kontrol *Trajectory Tracking* dengan *Model Predictive Control* pada sistem, pada berbagai kondisi yang mendekati kondisi aktual.

**BAB V PENUTUP**
Setelah proses pengujian dilakukan, disajikan beberapa kesimpulan yang disertai dengan saran terhadap kelanjutan Tugas Akhir ini.

## 1.7 Relevansi

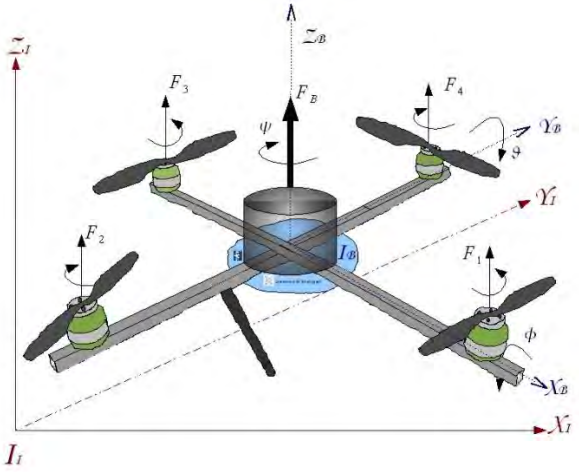
Hasil dari yang didapat pada tugas akhir ini diharapkan dapat menjadi referensi desain kontroler pada *quadcopter*, pengembangan dan perbandingan metode kontrol yang tepat untuk pengaturan gerak pada *quadcopter*.

*[Halaman ini sengaja dikosongkan]*

# BAB II
# TEORI PENUNJANG

## 2.1 *Quadcopter* [5][6]

*Quadcopter* yang juga dikenal sebagai quadrotor adalah helikopter dengan tersusun dari empat buah rotor yang ditempatkan dalam formasi persegi dengan jarak yang sama dari pusat massa *quadcopter* tersebut. *Quadcopter* dikendalikan dengan menyesuaikan kecepatan sudut dari rotor yang diputar putar oleh motor listrik. *Quadcopter* memiliki kemampan untuk melakukan beberapa gerakan seperti manuver bergerak maju atau mundur, tracking point (bergerak mengikuti suatu jalur), pendaratan (Landing), dan lepas landas secara vertikal (Vertical Take-Off). Dalam implementasinya, *quadcopter* difungsikan dalam suatu bidang x,y,z dengan titik pusat berada pada bagian tengah *quadcopter*. Ilustrasi sederhana bidang x,y, dan z pada *quadcopter* dapat dilihat pada Gambar 2.1.




**Gambar 2.1** Ilustrasi bidang x,y,dan z pada *quadcopter*

Sumbu x merupakan garis sejajar dengan bidang lengan depan *quadcopter*, sumbu y merupakan garis bidang sejajar dengan lengan

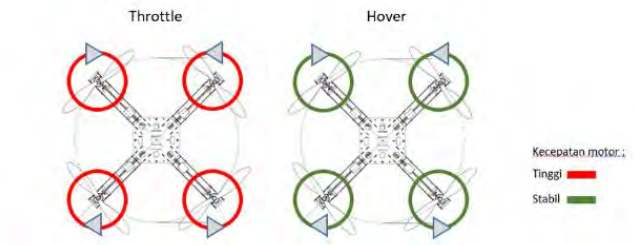
samping kanan *quadcopter*, dan sumbu z merupakan garis tegak lurus 90 derajat ke arah bawah *quadcopter*. Ketiga sumbu tersebut digunakan sebagai acuan gerak *quadcopter*.

### 2.1.1 Konsep Dasar *Quadcopter*

Dengan mengubah besaran kecepatan putaran keempat buah motor maka *quadcopter dapat* bergerak atas, bawah, maju, mundur, kiri, kanan, dan rotasi. Pergerakan tersebut lebih dikenal dengan istilah *pitch* (bergerak maju atau mundur), *roll* (bergerak kiri atau kanan), *yaw* (rotasi kiri atau rotasi kanan), *Hovering*/*Throtlle* (bergerak keatas). Penjelasan keempat gerakan dasar tersebut sebagai berikut :

**a. *Hovering/Throtlle***

Pada *hovering,* baling-baling depan dan belakang berputar searah jarum jam sedangkan baling-baling kiri dan kanan berputar berlawanan arah jarum jam dengan kecepatan putar yang sama, seperti pada Gambar 2.2. Perbedaan antara *hovering* dengan *throttle* adalah kecepatan motor pada saat *throttle* lebih besar daripada saat *hovering*. Karena *throttle* digunakan untuk melakukan lepas landas.

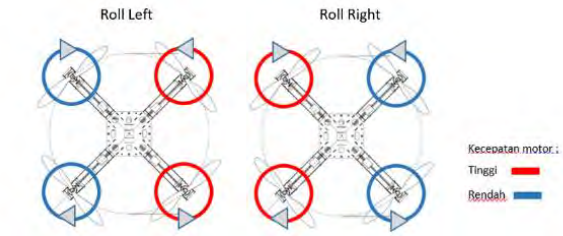



**Gambar 2.2** Pergerakan *Hovering/Throttle* pada *Quadcopter*

**b. *Roll***

Pergerakan Sudut *Roll* dapat dilihat pada Gambar 2.3. Pergerakan *Roll* dicapai dengan mengurangi atau meningkatkan kecepatan putar dari salah satu baling-baling kiri atau kanan dan melakukan perintah sebaliknya pada baling-baling yang berlawanan. Pergerakan ini akan menghasilkan manuver ke

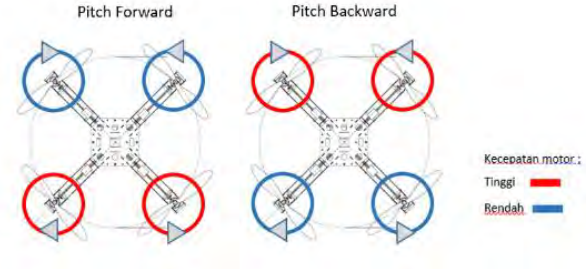
kanan dan kiri tergantung dari baling-baling mana yang diturunkan dan ditingkatkan kecepatannya.




**Gambar 2.3** Pergerakan *Roll* pada *Quadcopter*

**c. *Pitch***

Pola dari pergerakan ini memiliki prinsip yang sama dengan *roll* dan juga dilakukan untuk menghasilkan manuver seperti pada Gambar 2.4. Pada pergerakan *roll,* baling-baling yang diatur adalah kanan dan kiri untuk menghasilkan manuver ke kanan dan ke kiri. Sedangkan pada *Pitch*, pengaturan kecepatan baling-baling dilakukan pada baling-baling depan dan belakang untuk menghasilkan manuver ke depan dan ke belakang. Bila kecepatan baling-baling depan ditingkatkan/ diturunkan dan kecepatan baling-baling belakang diturunkan/ ditingkatkan, maka akan terjadi manuver ke belakang/ ke depan.

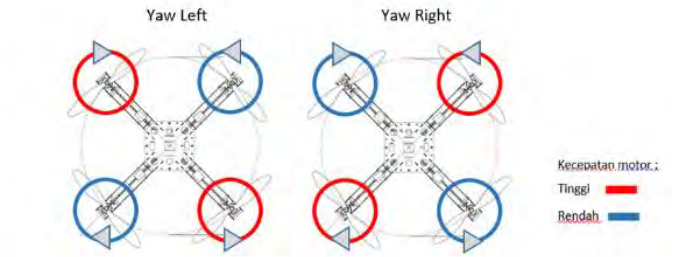



**Gambar 2.4** Pergerakan *Pitch* pada *Quadcopter*

**d. *Yaw***

*Yaw* adalah pergerakan yang dilakukan untuk memutar posisi quadrotor dengan inti tetap berada pada posisi yang sama. Pergerakan ini dicapai dengan mengurangi kecepatan 2 baling-

baling yang memiliki arah putar yang sama dan meningkatkan perputaran 2 baling-baling yang memiliki arah putar yang berlawanan dengan kedua baling-baling sebelumnya. Bila baling-baling kiri dan kanan kecepatanya diturunkan bersamaan dengan dinaikannya kecepatan baling-baling depan dan belakang, maka quadrotor berputar berlawanan arah jarum jam dengan inti sebagai poros. Begitu pula sebaliknya. Pola pergerakan dapat dilihat pada Gambar 2.5.




**Gambar 2.5** Pergerakan *Yaw* pada *Quadcopter*

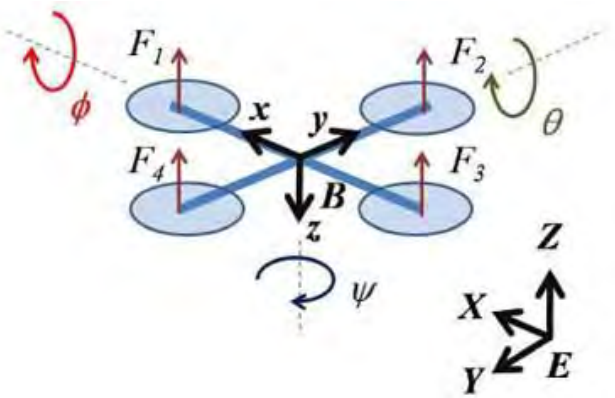
### 2.1.2 Pemodelan Kinematik *Quadcopter*

*Quadcopter* memiliki 6 *degree of freedom (DOF)*. Untuk mendeskripsikan gerakan dari 6 *DOF rigid-body* digunakan dua buah frame referensi yaitu *earth inertial reference* (*E-frame*) dan *body fixed reference* (*B-frame*). *B-frame* dan *E-frame* diasumsikan berada pada pusat gravitasi dari *Quadcopter* UAV, di mana sumbu z mengarah ke atas, seperti terlihat pada Gambar 2.6.

Persamaan gerak lebih mudah diformulasikan dalam *B-frame* karena matriks inersianya bersifat tidak berubah terhadap waktu, persamaan bisa disederhanakan dengan memanfaatkan keuntungan dari badan *Quadcopter* yang simetris, pengukuran lebih mudah dikonversikan kedalam *B-frame*, dan pengaturan gaya selalu diberikan pada *Body fixed frame.*

*E-frame* ($O_E$, $X_E$, $Y_E$, $Z_E$) dipilih sebagai acuan inersia benda. $Y_E$ menunjuk ke utara, $X_E$ menunjuk ke timur, $Z_E$ menunjuk ke atas (menjauhi bumi) dan $O_E$ menunjuk ke titik asal. *Frame* ini digunakan untuk menentukan posisi linier $\boldsymbol{\Gamma}_E$ (dalam meter) dan posisi angular $\boldsymbol{\Theta}_E$ (dalam radian) dari *quadcopter.*

*B-frame* menempel pada tubuh *quadcopter.* X menunjuk pada bagian depan *Quadcopter,* y menunjuk pada bagian kiri *quadcopter,* z menunjuk pada bagian atas, o menunjuk pada titik asal atau *origin.* Titik asal o digunakan untuk menunjukkan pusat dari *quadcopter*. Didalam *frame* ini ada kecepatan linier $V_B$ (m/s), kecepatan angular $\Omega_B$ (rad/s), gaya $F_B$ (N) dan torsi $\boldsymbol{\tau}_B$ (Nm). *E-frame* dan *B-frame* dapat dilihat pada Gambar 2.6.




**Gambar 2.6** *Earth-frame* dan *Body-frame* yang digunakan dalam permodelan *quadcopter*

Posisi linier *quadcopter* (ΓE) ditentukan dari koordinat vektor antara origin dan B-frame serta origin dari E-frame dengan memperhatikan E-frame dapat dilihat pada Persamaan (2.1).

$$\Gamma E = [\ X\ Y\ Z\ ]\ T \tag{2.1}$$

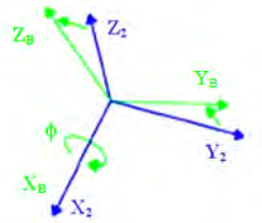
Posisi angular *quadcopter* (ΘE) ditentukan dari orientasi B-frame terhadap E-frame. Hal ini diperoleh dari tiga bentuk rotasi yang berurutan pada sumbu utama yang mengambil E-frame menjadi B-frame. Ketiga rotasi tersebut adalah *roll*, *pitch* dan *yaw*. Untuk mencari persamaan ketiga bentuk rotasi tersebut digunakan Euler Model.

$$\Theta E = [\ \phi\ \theta\ \psi\ ]\ T \tag{2.2}$$

Matriks rotasi RΘ diperoleh dari perkalian tiga rotasi matriks, seperti pada Gambar 2.7, Gambar 2.8 dan Gambar 2.9.

**a.** Rotasi sumbu X
Rotasi di sekitar sumbu x2 oleh sudut $\phi$ (*roll*) dengan matriks rotasi R($\phi$), ditunjukkan pada Gambar 2.7.

$$\begin{cases} x2 = xB` \\ y2 = yB` \cos\phi - zB` \sin\phi \\ z2 = zB` \cos\phi + yB` \sin\phi \end{cases} \quad (2.3)$$




**Gambar 2.7** Rotasi di Sekitar Sumbu $x_2$ oleh Sudut $\boldsymbol{\phi}$ (*Roll*)

Persamaan (2.3) dapat dibuat dalam bentuk matriks menjadi Persamaan (2.4).

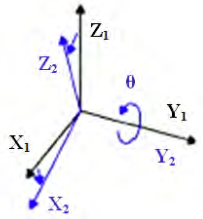
$$\begin{bmatrix} x_2` \\ y_2` \\ z_2` \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi & \cos\phi \end{bmatrix} \begin{bmatrix} x_B` \\ y_B` \\ z_B` \end{bmatrix} \quad (2.4)$$

Jadi persamaan matriks $\boldsymbol{R}(\theta)$, adalah sebagai berikut:

$$\mathbf{R}(\phi) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi & \cos\phi \end{bmatrix} \quad (2.5)$$

**b.** Rotasi sumbu Y
Rotasi di sekitar sumbu y1 oleh sudut $\theta$ (*pitch*) dengan matriks rotasi R($\theta$), ditunjukkan pada Gambar 2.8.

$$\begin{cases} x1 = x2` \cos\theta + z2` \sin\theta \\ y1 = y2` \\ z1 = z2` \cos\theta - x2` \sin\theta \end{cases} \quad (2.6)$$




**Gambar 2.8** Rotasi di Sekitar Sumbu $y_1$ oleh Sudut **θ** (*Pitch*)

Persamaan (2.6) dapat dibuat dalam bentuk matriks menjadi Persamaan (2.7).

$$\begin{bmatrix} x1` \\ y1` \\ z1` \end{bmatrix} = \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix} \begin{bmatrix} x1` \\ y1` \\ z1` \end{bmatrix} \quad (2.7)$$

Jadi persamaan matriks R(θ), adalah sebagai berikut:

$$\mathbf{R}(\theta) = \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix} \quad (2.8)$$

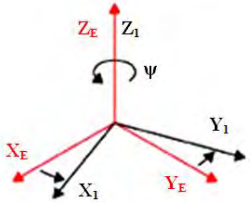
**c.** Rotasi sumbu Z

Rotasi di sekitar sumbu zE oleh sudut ψ (*yaw*) dengan matriks rotasi R(ψ), ditunjukkan pada Gambar 2.9.

$$\begin{cases} x_E = x_1' \cos\psi - y_1' \sin\psi \\ y_E = y_1' \cos\psi + x_1' \sin\psi \\ z_E = z_1 \end{cases} \quad (2.9)$$

Persamaan (2.9) dapat dibuat dalam bentuk matriks menjadi persamaan (2.10).

$$\begin{bmatrix} xE` \\ yE` \\ zE` \end{bmatrix} = \begin{bmatrix} \cos\psi & -\sin\psi & 0 \\ \sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x1` \\ y1` \\ z1` \end{bmatrix} \quad (2.10)$$




**Gambar 2.9** Rotasi di Sekitar Sumbu $Z_E$ oleh Sudut **ψ** (*Yaw*)

Jadi persamaan matriks R(ψ), adalah:

$$\mathbf{R}(\psi) = \begin{bmatrix} \cos\psi & -\sin\psi & 0 \\ \sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2.11}$$

Dari ketiga persamaan diatas, diperoleh persamaan matriks untuk gerak rotasi, yaitu:

$$R\Theta = R(\phi)\, R(\theta)\, R(\psi) \tag{2.12}$$

$$\mathbf{R\Theta} = \begin{bmatrix} c_\psi c_\theta & -s_\psi c_\phi + c_\psi s_\theta s_\phi & s_\psi s_\phi + c_\psi s_\theta s_\phi \\ s_\psi c_\theta & c_\psi c_\phi + s_\psi s_\theta c_\phi & -c_\psi s_\phi + s_\psi s_\theta c_\phi \\ -s_\theta & c_\theta s_\phi & c_\theta c_\phi \end{bmatrix} \tag{2.13}$$

Didalam persamaan sebelum dan sesudah ini, ada beberapa notasi yaitu ck = cos k, sk = sin k, tk = tan k. Seperti yang telah dinyatakan sebelumnya, kecepatan linear VB dan angular ωB diperlihatkan oleh body-fixed frame (B-frame). Komposisi vektornya disajikan pada Persamaan (2.14) dan (2.15).

$$VB = [u \quad v \quad w]^T \tag{2.14}$$

$$\omega B = [p \quad q \quad r]^T \tag{2.15}$$

Diperlukan kombinasi nilai linear dan angular untuk memberikan representasi yang lengkap dalam space. ξ merupakan komposisi dari vektor posisi linier ΓE (m) dan vektor posisi sudut ΘE (rad) terhadap E-frame seperti terlihat pada persamaan (2.16).

$$\xi = [\boldsymbol{\Gamma}^E \quad \boldsymbol{\Theta}^E]^T = [\, X \quad Y \quad Z \quad \phi \quad \theta \quad \psi]^T \tag{2.16}$$

**V** merupakan generalisasi dari vektor kecepatan linier *quadcopter* **V**B (m s-1) dan kecepatan angular *quadcopter* ωB (rad s-1) pada B-frame seperti terlihat pada persamaan dibawah ini:

$$\mathbf{V} = [\mathbf{V}^B \quad \boldsymbol{\omega}^B]^T = [\, u \quad v \quad w \quad p \quad q \quad r]^T \tag{2.17}$$

Hubungan antara kecepatan linear pada B-frame dan salah satu factor pada E-frame VE (atau $\boldsymbol{\Gamma}$E) [m/s] menyebabkan matriks rotasi $\mathbf{R}_{\boldsymbol{\Theta}}$ menurut Persamaan (2.18).

$$\mathbf{V}^{\mathrm{E}} = \dot{\boldsymbol{\Gamma}}^{\mathrm{E}} = \mathbf{R}_{\boldsymbol{\Theta}} \mathbf{V}^{\mathrm{B}} \tag{2.18}$$

Seperti pada kecepatan linear, hal tersebut juga berlaku untuk menghubungkan kecepatan angular pada E-frame (atau kecepatan Euler) **Θ**E [rad/s] ke B-frame $\boldsymbol{\omega}B$ berkat matriks transformasi TΘ. Persamaan (2.18) dan (2.19) menunjukkan hubungan tersebut.

$$\boldsymbol{\omega}^{\mathrm{B}} = \mathbf{T}_{\boldsymbol{\Theta}}^{-1} \dot{\boldsymbol{\Theta}}^{\mathrm{E}} \tag{2.19}$$

$$\dot{\boldsymbol{\Theta}}^{\mathrm{E}} = \mathbf{T}_{\boldsymbol{\Theta}} \boldsymbol{\omega}^{\mathrm{B}} \tag{2.20}$$

Matriks transformasi **T**Θ dapat ditetapkan dengan menggunakan kecepatan Euler dalam B-frame, dengan membalik pola perputaran sudut dari *roll*, *pitch* dan *yaw* seperti pada Persamaan (2.21).

$$\begin{bmatrix} p \\ q \\ r \end{bmatrix} = \begin{bmatrix} \dot{\phi} \\ 0 \\ 0 \end{bmatrix} + R(\phi)^{-1} \begin{bmatrix} 0 \\ \dot{\theta} \\ 0 \end{bmatrix} + R(\phi)^{-1} R(\theta)^{-1} \begin{bmatrix} 0 \\ 0 \\ \dot{\psi} \end{bmatrix} \tag{2.21}$$

$$\begin{bmatrix} p \\ q \\ r \end{bmatrix} = \boldsymbol{T}_{\boldsymbol{\Theta}}{}^{-1} \begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} \tag{2.22}$$

Dari persamaan (2.13) dan (2.19) maka diperoleh matriks transformasi dari bingkai bodi menuju bingkai bumi.

$$\boldsymbol{T_\Theta}^{-1} = \begin{bmatrix} 1 & 0 & -s_\theta \\ 0 & c_\phi & c_\theta s_\phi \\ 0 & -s_\phi & c_\phi c_\theta \end{bmatrix} \tag{2.23}$$

$$\boldsymbol{T_\Theta} = \begin{bmatrix} 1 & s_\phi t_\theta & c_\phi t_\theta \\ 0 & c_\phi & -s_\phi \\ 0 & s_\phi / c_\theta & c_\phi / c_\theta \end{bmatrix} \tag{2.24}$$

### 2.1.3 Pemodelan Dinamik *Quadcopter*

Pada bagian ini, akan dipelajari model dinamika *Quadcopter* UAV secara umum. Dinamika merupakan kebalikan dari kinematika, yang mempelajari gerak suatu objek tanpa memperhatikan apa penyebabnya. Sistem yang relatif ringan dan mampu terbang diudara, bentuk dinamik idealnya meliputi gyroscopic effect yang merupakan hasil resultan dari rotasi rigid-body pesawat diangkasa dan rotasi keempat propelernya. Model dinamik UAV menggunakan formulasi Euler-Lagrange.

$$\begin{cases} m\,\ddot{\boldsymbol{\Gamma}}^{\mathbf{E}} = \boldsymbol{F^E} \\ m\,\dot{\overbrace{\mathbf{R}_\Theta \mathbf{V}^{\mathbf{B}}}} = \boldsymbol{R_\Theta}\,\boldsymbol{F^B} \\ m\left(\mathbf{R}_\Theta \dot{\mathbf{V}}^{\mathbf{B}} + \dot{\mathbf{R}}_\Theta \mathbf{V}^{\mathbf{B}}\right) = \boldsymbol{F^B} \\ m\,\mathbf{R}_\Theta\left(\dot{\mathbf{V}}^{\mathbf{B}} + \boldsymbol{\omega}^{\mathbf{B}} \times \mathbf{V}^{\mathbf{B}}\right) = \mathbf{R}_\Theta \boldsymbol{F^B} \\ m\left(\dot{\mathbf{V}}^{\mathbf{B}} + \boldsymbol{\omega}^{\mathbf{B}} \times \mathbf{V}^{\mathbf{B}}\right) = \boldsymbol{F^B} \end{cases} \tag{2.25}$$

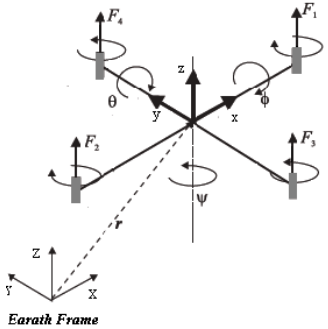
Persamaan (2.25) merupakan aksioma pertama Euler dari Hukum Kedua Newton, turunan dari komponen linear dari gerakan suatu benda. Dimana $m$ (kg) adalah massa *quadcopter*, $\ddot{\boldsymbol{\Gamma}}^{\mathbf{E}}$ (m s-2) adalah vektor percepatan linear *quadcopter* yang mengacu pada E-frame, $\boldsymbol{FE}$ (N) adalah vektor gaya *quadcopter* terhadap E-frame, $\dot{\mathbf{V}}^{\mathbf{B}}$ (m s-2) adalah percepatan linear *quadcopter* terhadap B-frame, dan $\dot{\mathbf{R}}_\Theta$ adalah turunan pertama matriks rotasi. Dan simbol × merupakan perkalian produk suatu vektor.

Berdasarkan aksioma kedua Euler dari Hukum Kedua Newton, dengan cara yang sama, turunan dari gerakan angular komponen dari suatu benda dapat dilihat pada Persamaan (2.26).

$$\begin{cases} I\ddot{\Theta}^{E} = \tau^{E} \\ I\ \dot{T_{\Theta}\omega^{B}} = T_{\Theta}\tau^{B} \\ I\dot{\omega}^{B} + \omega^{B}\, x\, (I\omega^{B}) = \tau^{B} \end{cases} \tag{2.26}$$

Pada Persamaan (2.26), $I$ (N m s2) adalah matriks inersia *quadcopter* (pada B-frame), $\ddot{\Theta}^{E}$ (rad s-2) adalah vektor percepatan sudut *quadcopter* terhadap E-frame, $\dot{\omega}^{B}$ (rad s-2) adalah vektor percepatan sudut *quadcopter* terhadap B-frame, dan $\tau E$ (N m) adalah torsi *quadcopter* terhadap E-frame. Dari Persamaan (2.25) dan (2.26), dapat dibentuk persamaan benda kaku dengan 6 derajat kebebasan (DOF). Persamaan (2.27) menunjukkan formulasi matriks dari dinamika sistem. *Quadcopter* memiliki empat buah motor yang menghasilkan gaya dorong seperti terlihat pada Gambar 2.10.

$$\begin{bmatrix} m\,I_{3\,x\,3} & 0_{3x3} \\ 0_{3x3} & I \end{bmatrix} \begin{bmatrix} \dot{V}^{B} \\ \dot{\omega}^{B} \end{bmatrix} + \begin{bmatrix} \omega^{B} \text{ x } (mV^{B}) \\ \omega^{B} \text{ x } (I\omega^{B}) \end{bmatrix} = \begin{bmatrix} F^{B} \\ \tau^{B} \end{bmatrix} \tag{2.27}$$




**Gambar 2.10** Gambar *E-frame* dan *B-frame Quadcopter* beserta empat gaya yang bekerja disetiap propeler

Dimana notasi I3x3 berarti matriks identitas 3x3. Dan 03x3 adalah matriks nol 3x3. Vektor gaya yang terjadi pada *quadcopter* dapat ditentukan berdasarkan persamaan (2.28).

$$\Lambda = [F^{B}\ \tau^{B}]^{T} = [F_x \quad F_y \quad F_z \quad \tau_x \quad \tau_y \quad \tau_z]^{T} \tag{2.28}$$

Persamaan (2.28) dapat juga ditulis dalam bentuk formulasi matriks seperti pada persamaan (2.29).

$$\boldsymbol{M_B}\dot{\boldsymbol{v}} + \boldsymbol{C_B}(\boldsymbol{v})\boldsymbol{v} = \boldsymbol{\Lambda} \tag{2.29}$$

Dimana $\dot{\boldsymbol{v}}$ adalah vektor percepatan *quadcopter* terhadap B-frame. $\boldsymbol{M}B$ adalah matriks inersia sistem dan $\boldsymbol{C}B$ adalah adalah matriks sentripetal Coriolis. Persamaan (2.30) menunjukkan matriks $\boldsymbol{M}B$.

$$\boldsymbol{M_B} = \begin{bmatrix} m\, I_{3x3} & 0_{3x3} \\ 0_{3x3} & I \end{bmatrix} = \begin{bmatrix} m & 0 & 0 & 0 & 0 & 0 \\ 0 & m & 0 & 0 & 0 & 0 \\ 0 & 0 & m & 0 & 0 & 0 \\ 0 & 0 & 0 & I_{XX} & 0 & 0 \\ 0 & 0 & 0 & 0 & I_{YY} & 0 \\ 0 & 0 & 0 & 0 & 0 & I_{ZZ} \end{bmatrix} \tag{2.30}$$

Dapat dilihat bahwa matriks $\boldsymbol{M}B$ adalah matriks diagonal, sedangkan matriks $\boldsymbol{C}B$ ditunjukkan oleh Persamaan (2.31).

$$\boldsymbol{C_B} = \begin{bmatrix} \mathbf{0_{3x3}} & -m\,\boldsymbol{S}(\mathbf{V_B}) \\ \mathbf{0_{3x3}} & -\boldsymbol{S}(\boldsymbol{I}\,\boldsymbol{\omega_B}) \end{bmatrix}$$

$$\boldsymbol{C_B} = \begin{bmatrix} 0 & 0 & 0 & 0 & m\,w & -m\,v \\ 0 & 0 & 0 & -m\,w & 0 & m\,u \\ 0 & 0 & 0 & m\,v & -m\,u & 0 \\ 0 & 0 & 0 & 0 & I_{ZZ}\,r & -I_{YY}\,q \\ 0 & 0 & 0 & -I_{ZZ}\,r & 0 & I_{XX}\,p \\ 0 & 0 & 0 & I_{YY}\,q & -I_{XX}\,p & 0 \end{bmatrix} \tag{2.31}$$

Pada persamaan ini, diadopsi operator skew-symmetric $\boldsymbol{S}$(.). Untuk vektor 3 dimensi k, matriks skew-symmetric dari k ($\boldsymbol{S}$(k)) ditunjukkan pada Persamaan (2.32).

$$\boldsymbol{S}(\boldsymbol{k}) = -\boldsymbol{S^T}(\boldsymbol{k}) = \begin{bmatrix} 0 & -k3 & k1 \\ k3 & 0 & -k1 \\ -k2 & k1 & 0 \end{bmatrix}, \boldsymbol{k} = \begin{bmatrix} k1 \\ k2 \\ k3 \end{bmatrix} \tag{2.32}$$

Matriks $\boldsymbol{\Lambda}$ dapat dibagi menjadi tiga komponen menurut kontribusi sifat dasar *quadcopter*. Yang pertama adalah vektor gravitasi $\boldsymbol{G}B(\boldsymbol{\xi})$ yang diperoleh dari percepatan gravitasi $g$ (m s-2). Percepatan gravitasi hanya berlaku pada persamaan linear *quadcopter*. Persamaan matriks $\boldsymbol{G}B(\boldsymbol{\xi})$ dituliskan pada Persamaan (2.33).

$$\boldsymbol{G_B(\xi)} = \begin{bmatrix} F_{GB} \\ 0_{3x1} \end{bmatrix} = \begin{bmatrix} \mathrm{R_\Theta}^{-1} F_{GE} \\ 0_{3x1} \end{bmatrix}$$

$$\boldsymbol{G_B(\xi)} = \begin{bmatrix} \mathbf{R_\Theta}^{\mathbf{T}} & \begin{bmatrix} 0 \\ 0 \\ -mg \end{bmatrix} \\ 0_{3x1} \end{bmatrix} = \begin{bmatrix} mg\ sin\ \theta \\ -mg\ cos\ \theta\ sin\ \phi \\ -\ mg\ cos\ \theta\ sin\ \phi \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{2.33}$$

di mana $\boldsymbol{FG}B$ (N) adalah vektor gaya gravitasi terhadap B-frame dan $\boldsymbol{FG}E$ (N) terhadap E-frame. $\mathbf{R\Theta}$ adalah matriks ortogonal, sehingga matriks inversnya sama dengan matriks transposnya.

Kontribusi yang kedua mengenai efek giroskopis yang dihasilkan oleh perputaran propeler. Selama dua propeler yang berhadapan berputar searah jarum jam dan dua propeler lainnya berputar berlawanan arah jarum jam, terjadi ketidak-seimbangan saat jumlah aljabar dari kecepatan motor tidak sama dengan nol. Jika *roll* dan *pitch* juga tidak nol, *quadcopter* akan mengalami torsi efek giroskopis yang ditunjukkan pada Persamaan (2.34).

$$\boldsymbol{O_B(v)\Omega} = \begin{bmatrix} 0_{3x1} \\ -\sum_{k=1}^{4} \boldsymbol{J_{TP}} \left( \boldsymbol{\omega}^{\mathbf{B}} \mathbf{x} \begin{bmatrix} \mathbf{0} \\ \mathbf{0} \\ \mathbf{1} \end{bmatrix} \right) (-\mathbf{1})^k \boldsymbol{\Omega_k} \end{bmatrix}$$

$$\boldsymbol{O_B(v)\Omega} = \begin{bmatrix} \mathbf{0_{3x1}} \\ \boldsymbol{J_{TP}} \begin{bmatrix} -\boldsymbol{q} \\ \boldsymbol{p} \\ \mathbf{0} \end{bmatrix} \boldsymbol{\Omega} \end{bmatrix}$$

$$\boldsymbol{O_B(v)\Omega} = \boldsymbol{J_{TP}} \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ -q & -q & -q & -q \\ -p & p & -p & p \\ 0 & 0 & 0 & 0 \end{bmatrix} \boldsymbol{\Omega} \tag{2.34}$$

$\boldsymbol{O}B$ adalah matriks giroskopis dari propeler dan JTP (N m s2) adalah momen inersia total di sekitar sumbu propeler. Efek giroskopis yang dihasilkan oleh putaran propeler hanya berhubungan dengan

persamaan angular, dan tidak mempengaruhi persamaan linear. $\Omega\,(\mathrm{rad\,s^{-1}})$ merupakan vektor kecepatan putar propeler, yang ditunjukkan oleh Persamaan (2.35).

$$\Omega = -\Omega_1 + \Omega_2 - \Omega_3 + \Omega_4, \qquad \boldsymbol{\Omega} = \begin{bmatrix} \Omega_1 \\ \Omega_2 \\ \Omega_3 \\ \Omega_4 \end{bmatrix} \tag{2.35}$$

di mana Ω1, Ω2, Ω3, dan Ω4 (rad s-1) adalah kecepatan propeler depan, kanan, belakang, dan kiri.

Kontribusi yang ketiga adalah perhitungan gaya dan torsi yang dihasilkan oleh pergerakan input. Pergerakan matriks $\boldsymbol{EB}$ dikali dengan $\boldsymbol{\Omega 2}$ untuk memperoleh vektor perpindahan $\mathbf{U}\mathrm{B}(\boldsymbol{\Omega})$. Efek aerodinamis (factor thrust $b$ (N s2) dan drag $d$ (N m s2)) berpengaruh pada gaya dan torsi yang dihasilkan. Persamaan vektor perpindahan pada dinamika *quadcopter* ditunjukkan pada Persamaan (2.36).

$$\boldsymbol{U_B}(\boldsymbol{\Omega}) = \boldsymbol{E_B}\boldsymbol{\Omega^2} = \begin{bmatrix} 0 \\ 0 \\ U_1 \\ U_2 \\ U_3 \\ U_4 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ b(\boldsymbol{\Omega}_1^{\,2} + \boldsymbol{\Omega}_2^{\,2} + \boldsymbol{\Omega}_3^{\,2} + \boldsymbol{\Omega}_4^{\,2}) \\ bl(-\boldsymbol{\Omega}_2^{\,2} + \boldsymbol{\Omega}_4^{\,2}) \\ bl(-\boldsymbol{\Omega}_1^{\,2} + \boldsymbol{\Omega}_3^{\,2}) \\ d(-\boldsymbol{\Omega}_1^{\,2} + \boldsymbol{\Omega}_2^{\,2} - \boldsymbol{\Omega}_3^{\,2} + \boldsymbol{\Omega}_4^{\,2}) \end{bmatrix} \tag{2.36}$$

di mana $l$ (m) adalah jarak antara pusat *quadcopter* dengan pusat propeler. $U1$, $U2$, $U3$, dan $U4$ adalah komponen vektor gerak. Hubungan komponen vektor tersebut dengan propeler dapat diperoleh dari perhitungan aerodinamis.

Dari persamaan sebelumnya, dapat diketahui konstanta matriks $\boldsymbol{EB}$ yang dikalikan dengan kuadrat dari kecepatan propeler $\boldsymbol{\Omega}2$ sehingga menghasilkan $\boldsymbol{UB}(\boldsymbol{\Omega})$. Persamaan (2.37) menunjukkan matriks $\boldsymbol{EB}$.

$$\boldsymbol{E_B} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ b & b & b & b \\ 0 & -bl & 0 & bl \\ -bl & 0 & bl & 0 \\ -d & d & -d & d \end{bmatrix} \tag{2.37}$$

Dari Persamaan (2.29), dinamika *quadcopter* dapat dibuat dengan mempertimbangkan ketiga kontribusi menurut Persamaan (2.38).

$$\boldsymbol{M_B\dot{v} + C_B(v)v = G_B(\xi) + O_B(v)\,\Omega + E_B\Omega^2} \tag{2.38}$$

Dengan mengatur ulang persamaan (2.38) diatas, turunan dari vektor kecepatan terhadap B-frame dapat dirumuskan sebagai berikut:

$$\boldsymbol{\dot{v} = M_B^{-1}\,(C_B(v)v + G_B(\xi) + O_B(v)\,\Omega + E_B\Omega^2)} \tag{2.39}$$

Persamaan (2.40) menunjukkan persamaan-persamaan sebelumnya bukan dalam bentuk matriks, melainkan dalam bentuk sistem persamaan.

$$\begin{cases} \dot{u} = (v\,r - w\,q) + g\,s_\theta \\ \dot{v} = (w\,p - u\,r) - g\,c_\theta s_\phi \\ \dot{w} = (u\,q - v\,p) - g\,c_\theta s_\phi + \frac{U_1}{m} \\ \dot{p} = \frac{I_{YY} - I_{ZZ}}{I_{XX}} q\,r - \frac{J_{TP}}{I_{XX}} q\,\Omega + \frac{U_2}{I_{XX}} \\ \dot{q} = \frac{I_{ZZ} - I_{XX}}{I_{YY}} p\,r - \frac{J_{TP}}{I_{YY}} p\,\Omega + \frac{U_3}{I_{YY}} \\ \dot{r} = \frac{I_{XX} - I_{YY}}{I_{ZZ}} p\,q - \frac{U_4}{I_{ZZ}} \end{cases} \tag{2.40}$$

Persamaan input kecepatan propeller (baling-baling) dapat dilihat dalam Persamaan (2.41).

$$\begin{cases} U_1 = b(\boldsymbol{\Omega}_1^{\,2} + \boldsymbol{\Omega}_2^{\,2} + \boldsymbol{\Omega}_3^{\,2} + \boldsymbol{\Omega}_4^{\,2}) \\ U_2 = bl(-\boldsymbol{\Omega}_2^{\,2} + \boldsymbol{\Omega}_4^{\,2}) \\ U_3 = bl(-\boldsymbol{\Omega}_1^{\,2} + \boldsymbol{\Omega}_3^{\,2}) \\ U_4 = d(-\boldsymbol{\Omega}_1^{\,2} + \boldsymbol{\Omega}_2^{\,2} - \boldsymbol{\Omega}_3^{\,2} + \boldsymbol{\Omega}_4^{\,2}) \\ \Omega = -\Omega_1 + \Omega_2 - \Omega_3 + \Omega_4 \end{cases} \tag{2.41}$$

Persamaan sistem dinamik *quadcopter* (2.40) diperoleh dari B-frame. Dalam pemodelannya, dibutuhkan persamaan dinamik sistem yang mencakup sistem hybrid yang disusun oleh persamaan linear dari E-frame dan persamaan angular dari B-frame.Hal ini dilakukan untuk mempermudah pengaturan. Persamaan-persamaan berikut akan disajikan

dalam kerangka "hybrid", atau H-frame. Persamaan (2.42) menunjukkan vektor kecepatan *quadcopter* yang digeneralisasi pada H-frame.

$$\dot{\boldsymbol{\xi}} = [\dot{\boldsymbol{\Gamma}}^{\mathbf{E}} \quad \boldsymbol{\omega}^{\boldsymbol{B}}]^{\boldsymbol{T}} = [\dot{X} \quad \dot{Y} \quad \dot{Z} \quad p \quad q \quad r]^{\boldsymbol{T}} \tag{2.42}$$

Dinamika sistem pada H-frame dapat dituliskan dalam bentuk martriks menurut Persamaan (2.43).

$$\boldsymbol{M_H}\dot{\boldsymbol{\xi}} + \boldsymbol{C_H}(\boldsymbol{\xi})\boldsymbol{\xi} = \boldsymbol{G_H} + \boldsymbol{O_H}(\boldsymbol{\xi})\,\boldsymbol{\Omega} + \boldsymbol{E_H}(\boldsymbol{\xi})\boldsymbol{\Omega^2} \tag{2.43}$$

Berikut ini akan ditentukan semua matriks dan vektor yang ditunjukkan oleh persamaan di atas. Matriks inersia sistem terhadap H-frame $\boldsymbol{M}H$ sama dengan matriks inersia sistem terhadap B-frame, dan ditentukan seperti pada Persamaan (2.44).

$$\boldsymbol{M_H} = \boldsymbol{M_B} = \begin{bmatrix} m & 0 & 0 & 0 & 0 & 0 \\ 0 & m & 0 & 0 & 0 & 0 \\ 0 & 0 & m & 0 & 0 & 0 \\ 0 & 0 & 0 & I_{XX} & 0 & 0 \\ 0 & 0 & 0 & 0 & I_{YY} & 0 \\ 0 & 0 & 0 & 0 & 0 & I_{ZZ} \end{bmatrix} \tag{2.44}$$

Namun, matriks sentripetal Coriolis terhadap H-frame $\boldsymbol{C}H(\boldsymbol{\zeta})$ tidak sama dengan matriks sentripetal Coriolis terhadap B-frame dan ditentukan seperti pada Persamaan (2.45).

$$\boldsymbol{C_H}(\boldsymbol{\xi}) = \begin{bmatrix} 0_{3x3} & 0_{3x3} \\ 0_{3x3} & -S(I\ \omega_{\mathrm{B}}) \end{bmatrix} \tag{2.45}$$

Vektor gravitasi terhadap H-frame $\boldsymbol{G}H$ dituliskan pada Persamaan (2.46). Dapat dilihat bahwa gravitasi mempengaruhi ketiga persamaan linear, namun lebih berpengaruh terhadap ketinggian *quadcopter*.

$$\boldsymbol{G_H} = \begin{bmatrix} F_G^E \\ 0_{3x1} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ -mg \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{2.46}$$

Efek giroskopis yang dihasilkan oleh putaran propeler tidak berubah karena hanya mempengaruhi persamaan angular yang mengacu kepada B-frame. Oleh karena itu, matriks giroskopis terhadap H-frame $\boldsymbol{O}H(\boldsymbol{\zeta})$ dibuat sama dengan Persamaan (2.47).

$$\boldsymbol{O_H}(\boldsymbol{\xi})\boldsymbol{\Omega} = \boldsymbol{O_B}(\boldsymbol{v})\,\boldsymbol{\Omega} = \begin{bmatrix} 0_{3x1} \\ J_{TP}\begin{bmatrix} -q \\ p \\ 0 \end{bmatrix}\Omega \end{bmatrix}$$

$$= \boldsymbol{J_{TP}} \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ q & -q & q & -q \\ -p & p & -p & p \\ 0 & 0 & 0 & 0 \end{bmatrix} \boldsymbol{\Omega} \qquad (2.47)$$

Matriks perpindahan terhadap H-frame $\boldsymbol{E}H(\boldsymbol{\xi})$ berbeda dengan perpindahan terhadap B-frame karena input $U1$ mempengaruhi semua persamaan linear dengan matriks rotasi $\mathbf{R\Theta}$. Perkalian produk antara matriks perpindahan dengan kuadrat kecepatan propeler ditunjukkan pada Persamaan (2.48).

$$\boldsymbol{E_H}(\boldsymbol{\xi})\boldsymbol{\Omega^2} = \begin{bmatrix} \boldsymbol{R_\Theta} & \boldsymbol{0_{3x3}} \\ \boldsymbol{0_{3x3}} & \boldsymbol{I_{3x3}} \end{bmatrix} \boldsymbol{E_B}\boldsymbol{\Omega^2}$$

$$= \begin{bmatrix} (\boldsymbol{s_\psi s_\phi} + \boldsymbol{c_\psi s_\theta c_\phi})\boldsymbol{U_1} \\ (-\boldsymbol{c_\psi s_\phi} + \boldsymbol{s_\psi s_\theta c_\phi})\boldsymbol{U_1} \\ (\boldsymbol{c_\theta c_\phi})\,\boldsymbol{U_1} \\ \boldsymbol{U_2} \\ \boldsymbol{U_3} \\ \boldsymbol{U_4} \end{bmatrix} \qquad (2.48)$$

Dengan menyusun kembali Persamaan (2.43), dapat ditemukan rumus untuk turunan vektor kecepatan yang digeneralisasi terhadap H-frame yang dapat dilihat pada persamaan (2.49).

$$\dot{\boldsymbol{\xi}} = \boldsymbol{M_H}^{-1}\,(\boldsymbol{C_H}(\boldsymbol{\xi})\boldsymbol{\xi} + \boldsymbol{G_H} + \boldsymbol{O_H}(\boldsymbol{\xi})\,\boldsymbol{\Omega} + \boldsymbol{E_H}(\boldsymbol{\xi})\boldsymbol{\Omega^2}) \qquad (2.49)$$

### 2.1.4 Model Matematika *Quadcopter*

Dari analisis kinematika dan dinamika diperoleh Persamaan model matematika dari *quadcopter* seperti pada Persamaan (2.50).

$$\begin{cases} \ddot{X} = \frac{U_1}{m}(\sin\psi \sin\phi + \cos\psi \sin\theta \cos\phi) \\ \ddot{Y} = \frac{U_1}{m}(-\cos\psi \sin\phi + \sin\psi \sin\theta \cos\phi) \\ \ddot{Z} = -g + \frac{U_1}{m}(\cos\theta \cos\phi) \\ \dot{p} = \frac{I_{YY}-I_{ZZ}}{I_{XX}} q\, r - \frac{J_{TP}}{I_{XX}} q\, \Omega + \frac{U_2}{I_{XX}} \\ \dot{q} = \frac{I_{ZZ}-I_{XX}}{I_{YY}} p\, r - \frac{J_{TP}}{I_{YY}} p\, \Omega + \frac{U_3}{I_{YY}} \\ \dot{r} = \frac{I_{XX}-I_{YY}}{I_{ZZ}} p\, q - \frac{U_4}{I_{ZZ}} \end{cases} \quad (2.50)$$

Dengan melihat secara sederhana pada Persamaan (2.50), posisi pada sumbu Z, dan posisi sudut *roll*, *pitch*, *yaw* dapat dikontrol secara langsung, berturut-turut dengan menggunakan U1, U2, U3,dan U4. Kontrol pada posisi maju (X), dan menyamping (Y) dapat dilakukan dengan mengatur sudut *pitch* dan *roll* dengan syarat gaya angkat (U1) tidak sama dengan nol.

Nilai input dari *quadcopter* merupakan gaya angkat tiap propeler, yang telah dimodelkan secara teoritis dalam persamaan (2.41), adalah sebagai berikut:

$$\begin{cases} U_1 = b(\Omega_1^{\,2} + \Omega_2^{\,2} + \Omega_3^{\,2} + \Omega_4^{\,2}) \\ U_2 = bl(-\Omega_2^{\,2} + \Omega_4^{\,2}) \\ U_3 = bl(-\Omega_1^{\,2} + \Omega_3^{\,2}) \\ U_4 = d(-\Omega_1^{\,2} + \Omega_2^{\,2} - \Omega_3^{\,2} + \Omega_4^{\,2}) \\ \Omega = -\Omega_1 + \Omega_2 - \Omega_3 + \Omega_4 \end{cases} \quad (2.51)$$

## 2.2 *Feedback Linearization* [7]

Suatu sistem yang kompleks memiliki hubungan antara besaran-besaran fisik dalam sistem bersifat nonlinear. Untuk menyelesaikan persoalan nonlinear, dapat digunakan dengan menggunakan pendekatan persamaan linear. Salah satu kharakteristik dari sistem nonlinear adalah ketergantungan perilaku respon sistem pada besaran dan input dari sistem.

Quadcopter merupakan sistem yang memiliki hubungan antar variable bersifat nonlinear. Pada Persamaan (2.50) terlihat bahwa

persamaan tersebut terdapat hubungan antar variable bersifat nonlinear karena terdapat fungsi trigonometri. Untuk dapat melakukan proses kontrol yang menggunakan kontroler linear, maka model matematika tersebut harus dilinearisasi. Teori linearisasi digunakan untuk mengatasi permasalahan sistem nonlinear agar dapat didekati dengan sistem linear.

Pada penelitian ini digunakan teknik linearisasi *feedback* linearization di mana merupakan salah satu teknik linearisasi dengan melakukan modifikasi pada variabel yang dikontrol sehingga dapat didekati dengan sistem linear. Diagram blok *feedback* linearization dapat dilihat pada Gambar 2.11. Misalkan suatu sistem mempunyai persamaan state seperti Persamaan (2.52).

$$\dot{x} = f(x) + g(x)u$$
$$y = h(x) \tag{2.52}$$

Jika Persamaan (2.52) ditambahkan dengan suatu fungsi yang bernilai 0 maka Persamaan (2.52) tidak berubah. Sehingga Persamaan 2.52 menjadi Persamaan (2.53).

$$\dot{x} = f(x) + g(x)u + Ax - Ax + Bu^* - Bu^* \tag{2.53}$$

Agar Persamaan (2.53) menjadi persamaan state linier maka persamaan $f(x) + g(x)u - Ax - Bu^*$ disama dengankan dengan nol. Sehingga didapatkan Persamaan (2.54).

$$u = g(x)^{-1}(Ax + Bu^* - f(x)) \tag{2.54}$$

## 2.3 Kontroler

Dalam sebuah sistem kontrol, kontroler mempunyai kontribusi yang besar terhadap perilaku sistem. Pada prinsipnya hal itu disebabkan oleh tidak dapat diubahnya komponen penyusun sistem tersebut. Artinya, karakteristik *plant* harus diterima sebagaimana adanya, sehingga perubahan perilaku sistem hanya dapat dilakukan melalui penambahan suatu sub sistem, yaitu kontroler [8].

Salah satu tugas kontroler adalah meminimalkan sinyal kesalahan, yaitu perbedaan antara sinyal set point dan sinyal aktual. Hal ini sesuai dengan tujuan sistem kontrol yaitu memperoleh sinyal aktual yang senantiasa (diinginkan) sama dengan sinyal setpoint. Semakin cepat reaksi sistem (sinyal aktual) mengikuti sinyal setpoint dan semakin kecil

kesalahan yang terjadi, semakin baiklah kinerja sistem kontrol yang diterapkan.

Dalam Sub bab ini akan dibahas teori tentang kontroler yang digunakan dalam penelitian, yaitu kontroler proporsional, integral dan deferensial dan *Model Predictive Control* (MPC).

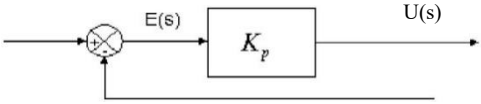
### 2.3.1 Kontroler Proporsional, Integral, dan *Derivative* (PID) [9][10]

Didalam suatu sistem kontrol terdapat beberapa macam aksi kontrol, diantaranya yaitu aksi kontrol proporsional, aksi kontrol integral dan aksi kontrol derivative. Masing-masing aksi kontrol ini mempunyai keunggulan-keunggulan tertentu, dimana aksi kontrol proporsional mempunyai keunggulan rise time yang cepat, aksi kontrol integral mempunyai keunggulan untuk memperkecil *error* ,dan aksi kontrol derivative mempunyai keunggulan untuk memperkecil error atau meredam overshot/undershot. Untuk itu agar dapat menghasilkan output dengan risetime yang cepat dan error yang kecil dapat dengan menggabungkan ketiga aksi kontrol ini menjadi aksi kontrol PID. Parameter pengontrol Proporsional Integral derivative (PID) selalu didasari atas tinjauan terhadap karakteristik yang di atur (*plant*). Sebelum dikontrol dengan kontroler PID, perilaku *plant* yang akan dikontrol harus di ketahui terlabih dahulu sebelum pencarian parameter PID dilakukan.

#### 2.3.1.1 Kontroler Proposional

Kontroler proposional memiliki keluaran yang sebanding atau proposional dengan besarnya sinyal kesalahan (selisih antara besaran yang di inginkan dengan harga aktualnya). Secara lebih sederhana dapat dikatakan bahwa keluaran pengontrol proporsional merupakan perkalian antara konstanta proposional dengan masukannya. Perubahan pada sinyal masukan akan segera menyebabkan sistem secara langsung mengeluarkan output sinyal sebesar konstanta pengalinya.

Gambar 2.11 menunjukkan blok diagram yang menggambarkan hubungan antara besaran setting, besaran aktual dengan besaran keluaran pengontrol proporsional. Sinyal keasalahan (error) merupakan selisih antara besaran setting dengan besaran aktualnya. Selisih ini akan mempengaruhi pengontrol, untuk mengeluarkan sinyal positif (mempercepat pencapaian harga setting) atau negatif (memperlambat tercapainya harga yang diinginkan).



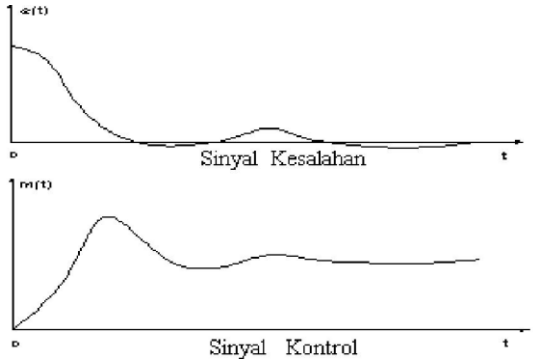

**Gambar 2.11** Diagram blok kontroler proporsional

### 2.3.1.2 Kontroler Integral

Pengontrol integral berfungsi menghasilkan respon sistem yang memiliki kesalahan keadaan stabil nol. Jika sebuah *plant* tidak memiliki unsur integrator (1/s), pengontrol proposional tidak akan mampu menjamin keluaran sistem dengan kesalahan keadaan stabilnya nol. Dengan pengontrol integral, respon sistem dapat diperbaiki, yaitu mempunyai kesalahan keadaan stabilnya nol.

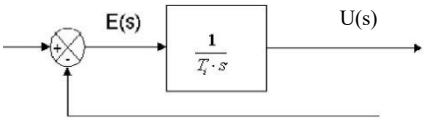
Pengontrol integral memiliki karaktiristik seperti halnya sebuah integral. Keluaran sangat dipengaruhi oleh perubahan yang sebanding dengan nilai sinyal kesalahan. Keluaran pengontrol ini merupakan penjumlahan yang terus menerus dari perubahan masukannya. Kalau sinyal kesalahan tidak mengalami perubahan, keluaran akan menjaga keadaan seperti sebelum terjadinya perubahan masukan.

Sinyal keluaran pengontrol integral merupakan luas bidang yang dibentuk oleh kurva kesalahan penggerak. Sinyal keluaran akan berharga sama dengan harga sebelumnya ketika sinyal kesalahan berharga nol. Gambar 2.12 menunjukkan contoh sinyal kesalahan yang dimasukan ke dalam pengontrol integral dan keluaran pengontrol integral terhadap perubahan sinyal kesalahan tersebut.




**Gambar 2.12** Kurva sinyal kesalahan e(t) terhadap t pada pembangkit kesalahan nol.

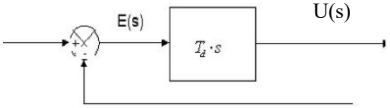
Gambar 2.13 menunjukkan blok diagram antara besaran kesalahan dengan keluaran suatu pengontrol integral.




**Gambar 2.13** Diagram blok kontroler integral

### 2.3.1.3 Kontroler Derivative

Keluaran pengontrol Derivative memiliki sifat seperti halnya suatu operasi differensial. Perubahan yang mendadak pada masukan pengontrol, akan mengakibatkan perubahan yang sangat besar dan cepat. Gambar 2.14 menunjukkan blok diagram yang menggambarkan hubung1an antara sinyal kesalahan dengan keluaran pengontrol.

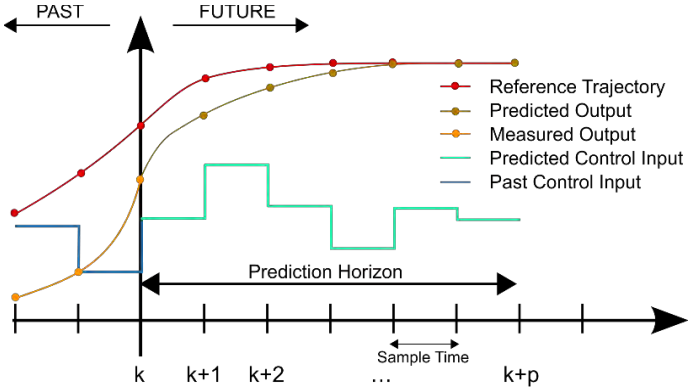



**Gambar 2.14** Diagram blok kontroler derivative

### 2.3.2 *Model Predictive Control* (MPC) [11]

Tujuan utama dari sebuah *Model Predictive Control* (MPC) adalah untuk menghitung trayektori dari sinyal kontrol *u* (*manipulated variable*)

yang akan datang untuk mengoptimalkan perilaku yang akan datang (*future behavior*) pada sinyal *output y* pada sebuah *plant* berdasarkan pada nilai pengukuran saat ini dan prediksi dari nilai *output* yang akan datang. Objektif dari kontroler MPC adalah untuk menentukan nilai sinyal kontrol (*sequence of control moves*) sehingga nilai *output* yang diprediksi akan mendekati nilai *setpoint* dengan optimal. Pada Gambar 2.15, dapat dilihat susunan dari nilai *output* saat ini (*actual output*) *y*, nilai *output* terprediksi (*predicted output*) $y\hat{}$, dan *manipulated input* atau sinyal kontrol *u*.




**Gambar 2.15** Konsep dari Kontroler *Model Predictive Control*

Pada setiap waktu *sampling k*, kontroler MPC menghitung himpunan dari nilai *M* atau *control horizon* (selanjutnya disebut $N_c$) dari *input* $\{u(k+i-1), i=1, 2,..., M\}$. Nilai *input* akan ditahan pada nilai konstan setelah *M* langkah didalam sinyal kontrol tersebut. Nilai *input* akan dihitung sedemikian sehingga nilai himpunan dari *P* keluaran atau *output* terprediksi $\{y(k+i), i = 1, 2,..., P\}$ mencapai nilai *setpoint* yang diinginkan. *P* merupakan nilai dari *prediction horizon* (selanjutnya disebut $N_p$) pada kontroler MPC. Perhitungan nilai kontrol pada kontroler MPC dihitung berdasarkan nilai optimal dari suatu fungsi objektif atau indeks performansi *J*.

### 2.3.2.1 Model *State-Space* dengan *Embedded Integrator*

Sistem *Model Predictive Control* didesain berdasarkan oleh model matematika *plant*. Model *plant* yang akan digunakan untuk desain system

*control* dirubah menjadi model *state space* yang diperlukan untuk memprediksi respon kedepan yang diwakilkan oleh variable saat ini. Untuk mempermudah, diasumsikan *plant* merupakan sistem single-input dan single output yang dapat dideskripsikan sebagai berikut :

$$x_m(k+1) = A_m x_m(k) + B_m u(k) \tag{2.55}$$

$$y(k) = C_m x_m(k) + D_m u(k) \tag{2.56}$$

di mana u adalah variabel manipulasi, y adalah variabel ouput dan xm adalah variabel state. Dikarenakan prinsip dari receding *horizon control*, dimana state saat ini dibutuhkan untuk menghitung prediksi dan kontrol, maka dapat diasumsikan bahwa input u(k) tidak dapat mempengaruhi output y(k) pada waktu yang sama. Oleh karena itu Dm dapat diabaikan, sehingga Persamaan (2.55) dan Persamaan (2.56) dapat ditulis sebagai berikut :

$$x_m(k+1) = A_m x_m(k) + B_m u(k) \tag{2.57}$$

$$y(k) = C_m x_m(k) \tag{2.58}$$

Kedua sisi dari Persamaan (2.57) dilakukan operesasi beda, sehingga persamaannya menjadi sebagai berikut :

$$\begin{aligned} x_m(k+1) - x_m(k) = A_m\big(x_m(k) - x_m(k-1)\big) \\ +B_m\big(u(k) - u(k-1)\big) \end{aligned} \tag{2.59}$$

Persamaan beda variable state dapat dinotasikan sebagai berikut :

$$\Delta x_m(k+1) = x_m(k+1) - x_m(k) \tag{2.60}$$

$$\Delta x_m(k) = x_m(k) - x_m(k-1) \tag{2.61}$$

*S*edangkan untuk persamaan beda vaiabel *control* dapat dinotasikan sebagai berikut :

$$\Delta u(k) = u(k) - u(k-1) \tag{2.62}$$

$$\Delta x_m(k+1) = A_m \Delta x_m(k) + B_m \Delta u(k) \tag{2.63}$$

Untuk menghubungkan $\Delta x_m(k)$ ke output $y(k)$ maka dibentuk vektor variabel state baru yaitu,

$$x(k) = [\Delta x_m(k)^T \; y(k)]^T \tag{2.64}$$

$$y(k+1) - y(k) = C_m\big(x_m(k+1) - x_m(k)\big) = C_m \Delta x_m(k)$$

Atau

$$y(k+1) - y(k) \; = C_m A_m \Delta x_m(k) + C_m B_m \Delta u(k) \tag{2.65}$$

Persamaan (2.64) dan Persamaan (2.65) digabungkan membentuk model *state space*,

$$\overbrace{\begin{bmatrix} \Delta x_m(k+1) \\ y(k+1) \end{bmatrix}}^{x(k+1)} = \overbrace{\begin{bmatrix} A_m & o_m^T \\ C_m A_m & 1 \end{bmatrix}}^{A} \begin{bmatrix} \Delta x_m(k) \\ y(k) \end{bmatrix} + \overbrace{\begin{bmatrix} B_m \\ C_m B_m \end{bmatrix}}^{B} \Delta u(k) \tag{2.66}$$

$$y(k) = \overbrace{\begin{bmatrix} o_m & 1 \end{bmatrix}}^{C} \begin{bmatrix} \Delta x_m(k) \\ y(k) \end{bmatrix} \tag{2.67}$$

dimana $o_m = \overbrace{[0\; 0 \ldots 0]}^{n_1}$, sedangkan matriks A,B, dan C disebut dengan matriks augmented yang akan digunakan untuk desain dari predictive *control*.

#### *2.3.2.2 Prediction of State and Output Variables*

Setelah mendapatkan augmented model, langkah selanjutnya merupakan menghitung nilai output terprediksi dan variabel kontrol yang akan datang. Variabel kontrol yang akan datang dapat dinotasikan seperti pada Persamaan (2.68).

$$\Delta u(k_i), \Delta u(k_i + 1), \ldots, \Delta u(k_i + N_c - 1) \tag{2.68}$$

$k_i$ merupakan instan sampling sedangkan Nc merupakan nilai *control horizon*, yaitu jumlah langkah kontrol berkelanjutan yang diterapkan dan diprediksi oleh kontroler MPC dalam sebuah sampling time. Selain itu, variabel output terprediksi dapat diperkirakan dan diprediksi dalam jumlah sampel Np, dimana Np merupakan nilai prediction *horizon*. Adapun variabel output terprediksi dapat dituliskan dalam Persamaan (2.68).

$$x(k_i + 1|k_i), \ldots, x(k_i + m|k_i), \ldots, x\left(k_i + N_p|k_i\right) \tag{2.69}$$

Dimana $x(k_i + 1|k_i)$ merupakan variabel state yang terprediksi saat $k_i + m$ dengan diberikan informasi *plant* saat ini $x(k_i)$, nilai Nc harus lebih kecil atau sama dengan nilai Np. Setelah itu, nilai output terprediksi dan variabel kontrol yang akan datang dapat dihitung dengan menggunakan Persamaan (2.70).

$$Y = Fx(k_i) + \Phi\Delta U \tag{2.70}$$

di mana matriks F, Φ, dan $\Delta U$ dapat diformulasikan sebagai berikut:

$$\Delta U = [u(k_i)\ \Delta u(k_i + 1) \ldots.\ \Delta u(k_i + N_c - 1)]^T \tag{2.71}$$

$$F = \begin{bmatrix} CA \\ CA^2 \\ CA^3 \\ \vdots \\ CA^{N_p} \end{bmatrix} \tag{2.72}$$

$$\Phi = \begin{bmatrix} CB & 0 & 0 & \cdots & 0 \\ CAB & CB & 0 & \cdots & 0 \\ CA^2B & CAB & CB & \cdots & 0 \\ \vdots & & & & \\ CA^{N_p} & CA^{N_p} & CA^{N_p} & \cdots & CA^{N_p} \end{bmatrix} \tag{2.73}$$

#### 2.3.2.3 Indeks performansi kontroler MPC

Dalam sebuah kontroler MPC, diperlukan proses optimasi yang mempunyai objektif kontrol untuk meminimalkan error yang terbentuk dari selisih nilai referensi dengan nilai keluaran dari *plant*. Optimasi

tersebut dilakukan dengan mendeskripsikan sebuah nilai dan parameter indeks performansi J yang merefleksikan objektif kontrol dari kontroler MPC. Indeks performansi tersebut dapat didefinisikan sebagai berikut:

$$R_s^T = \overbrace{[1\ 1 \dots 1]}^{N_p} r(k_i) \tag{2.74}$$

$$R_s = \overbrace{[1\ 1 \dots 1]^T}^{N_p} r(k_i) = \overline{R_s} r(k_i) \tag{2.75}$$

$$J = (R_s - Y)^T (R_s - Y) + \Delta U^T \bar{R} \Delta U \tag{2.76}$$

$R_s$ merupakan vektor yang berisi informasi sinyal referensi sinyal set point yang dinotasikan pada persamaan (2.75). Persamaan $(R_s - Y)^T (R_s - Y)$ pada persamaan (2.76) indeks performansi J mempunyai tujuan untuk meminimalkan error yang terjadi antara nilai output yang terpredisi dengan set point yang diberikan. Sedangkan persamaan $\Delta U^T \bar{R} \Delta U$ untuk merefleksikan seberapa besar nilai $\Delta U$ yang akan dihasilkan ketika fungsi objektif indeks performansi J dibuat sekecil mungkin. Matriks $\bar{R}$ adalah matriks diagonal yang berbentuk $\bar{R} = r_w I_{N_p \times N_p} (r_w \geq 0)$ digunakan sebagai parameter tuning kontroler MPC. Variabel $r_w$ merupakan tuning parameter untuk performa *closed loop* system pada kontroler MPC. Pada saat nilai $r_w = 0$, indeks performansi J akan mempunyai objektif untuk meminimalkan nilai error $(R_s - Y)^T (R_s - Y)$ sekecil mungkin tanpa memperdulikan seberapa besar nilai $\Delta U$. Untuk kasus dimana nilai rw dibuat semakin besar, indeks performansi J pada Persamaan (2.76) akan diterjemahkan ke dalam situasi dimana kita akan meminimalkan nilai error $(R_s - Y)^T (R_s - Y)$ secara hati-hati dengan mempertimbangkan seberapa besar nilai $\Delta U$ yang akan dihasilkan kontroler MPC. Indeks performansi dapat juga dinotasikan sebagai Persamaan (2.76).

$$J = \big(R_s - Fx(k_i)\big)^T \big(R_s - Fx(k_i)\big) - 2\Delta U^T \Phi^T \big(R_s - Fx(k_i)\big) + \Delta U^T (\Phi^T \Phi + \bar{R}\,) \Delta U \tag{2.77}$$

Setelah itu, Persamaan (2.77) diturunkn terhadap $\Delta U$ sehingga persamaannya menjadi seperti Persamaan (2.78)

$$\frac{\partial J}{\partial \Delta U} = \left(R_s - Fx(k_i)\right) + 2(\Phi^T\Phi + \bar{R}\,)\Delta U \tag{2.78}$$

Kondisi yang dibutuhkan untuk meminimalkan indeks performansi dicari pada kondisi sebagai berikut :

$$\frac{\partial J}{\partial \Delta U} = 0 \tag{2.79}$$

Dengan mensubtitusi Persamaan (2.78) dengan Persamaan (2.79) maka dapat disimpulkan solusi optimal dari sinyal *control* pada kontroler MPC sebagai Persamaan (2.80)

$$\Delta U = (\Phi^T\Phi + \bar{R}\,)^{-1}\Phi^T\left(R_s - Fx(k_i)\right) \tag{2.80}$$

#### *2.3.2.4 Closed loop Control System*

Nilai optimal parameter pada vektor $\Delta U$mengandung sinyal *control* $\Delta u(k_i), \Delta u(k_i+1), \ldots, \Delta u(k_i+N_c-1)$ tetapi hanya sampel pertama dari urutan atau sequence yang dapat diimplementasikan dan mengabaikan urutan selanjutnya. Prinsip tersebut disbut dengan *Receding Horizon Control*(RHC). Ketika periode sampling selanjutnya datang, nilai pengukuran yang paling baru diambil dari *state vector* $(k_i+1)$ untuk perhitungan sinyal kontrol yang baru. Prosedur ini terus berlanjut pada kondisi *real time* untuk memenuhi prinsip RHC. Oleh karena itu, sinyal *control* yang didapatkan dari Persamaan (2.80) dapat ditulis ulang seperti pada Persamaan (2.81).

$$\Delta u = (\Phi^T\Phi + \bar{R}\,)^{-1}\left(\Phi^T R_s r(k_i) - \Phi^T Fx(k_i)\right)$$

Atau

$$\Delta u = K_y r(k_i) - K_{mpc}x(k_i) \tag{2.81}$$

Dimana $K_y$ adalah baris pertama dari matriks $(\Phi^T\Phi + \bar{R}\,)^{-1}\Phi^T R_s$ dan $K_{mpc}$ adalah baris pertama dari matriks $(\Phi^T\Phi + \bar{R}\,)^{-1}\Phi^T F$. Persamaan (2.81) adalah bentuk standar dari linear time-invariant state *feedback control*. *Gain* dari state *feedback control* adalah $K_{mpc}$. Untuk mencari persamaan system *closed loop* pada kontroler MPC, digunakan augmented model. Seperti pada Persamaan (2.82).

$$x(k+1) = Ax(k) + B\Delta u(k) \tag{2.82}$$

Sistem *closed loop* seperti pada Gambar 2.19 dapat dicari dengan mensubtitusi Persamaan (2.81) ke dalam Persamaan (2.82) dan menggandi indeks $k_i$ ke dalam $k$ sebagai mana Persamaan (2.83).

$$x(k+1) = Ax(k) - BK_{mpc}x(k) + BK_y r(k)$$

atau

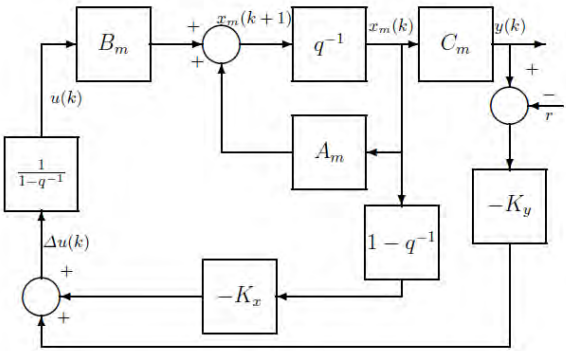
$$x(k+1) = \left(A - BK_{mpc}\right)x(k) + BK_y r(k) \tag{2.83}$$

Nilai eigenvalues pada system closed-loop dapat diteukan dengan mengamati persamaan karakteristik closed-loop seperti pada Persamaan (2.84).

$$det\left[\lambda I - (A - BK_{mpc})\right] = 0 \tag{2.84}$$

Dikarenakan struktur unik dari matriks A dan C, kolom terakhir dari matriks F identik dengan matriks $\overline{R_s}$, yaitu $[1\ 1 \dots 1]^T$. Oleh karena itu, *gain* $K_y$ identik dengan elemen terakhir pada *gain* $K_{mpc}$.

Perlu dicatat bahwa nilai vektor state variable $x(k) = [\Delta x_m(k)^T\ y(k)]^T$ dan dengan definisi $K_y$sehingga dapat mendeklarasikan bahwa *gain* $K_{mpc} = [K_x \quad K_y]$, dimana $K_x$ mendefinisikan hubungan *feedback gain* vector dengan $\Delta x_m(k)$. Sedangkan $K_y$ mendefinisikan hubungan antara *gain* dengan $y(k)$. Oleh karena itu, blok diagram sistem *closed-loop* pada kontroler MPC seperti pada Gambar 2.16. Notasi $q^{-1}$ merupakan notasi *backward shift operator* dan $\frac{1}{1-q^{-1}}$ menotasikan *discrete-time integrator*.




**Gambar 2.16** Sistem *Closed loop* Kontroler MPC

# BAB III
# PERANCANGAN SISTEM

Perancangan-perancangan simulasi sistem dari *quadcopter* yang terdiri atas Desain *quadcopter*, identifikasi sistem dan perancangan kontroler yang meliputi desain mekanik *quadcopter*, desain elektronik *quadcopter*, parameter *quadcopter*, model matematika hasil identifikasi, perancangan kontroler PID maupun kontroler MPC.

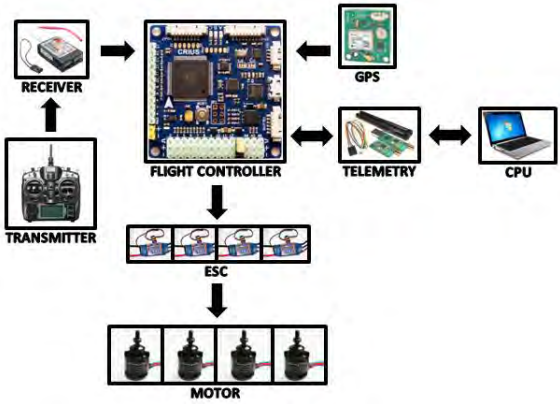
## 3.1 Desain *Quadcopter*

*Quadcopter* yang digunakan adalah *quadcopter* rakitan/*hobby* yang terdapat pada Laboratorium AJ 204 Teknik Elektro. Desain mekanik pada *quadcopter hobby* LAB AJ 204 seperti Gambar 3.1. Desain mekanik pada *quadcopter* dibuat simetris, seimbang dan ringan. Kesimetrisan *quadcopter* dapat ditunjukkan dengan nilai momen inersia pada sumbu translasi yang memiliki nilai yang sama.

**Gambar 3.1** Desain Mekanik *Quadcopter hobby* LAB AJ 204

Sedangkan desain eletronikknya seperti Gambar 3.2. Desain elektronika pada *quadcopter* rakitan ini terdiri dari sistem kontroler yang berupa Arduino pilot Mega 2.6 yang dilengkapi dengan ATMega 2560 sebagai mikrokontroler dari flight *control*ler. Selain itu juga dilengkapi

dengan GPS yang digunakan untuk mengetahui posisi *quadcopter* dan *telemetry* 433 Mhz yang digunakan sebagai komunikasi data.




**Gambar 3.2** Desain Sistem Elektronka *Quadcopter hobby* LAB AJ 204

## 3.2 Identifikasi Sistem

Dari desain mekanik maupun elektronik *quadcopter* dapat diperoleh parameter-parameter yang digunakan untuk melengkapi model matematika *quadcopter*. Dimulai dari menghitung massa keseluruhan dari *quadcopter* dengan alat pengukur massa. Momen inersia sumbu X, sumbu Y dan sumbu Z didapat dari identifikasi input-output sistem dengan cara menerbangkan *quadcopter* lalu didapatkan data keluaran dari sensor yaitu berupa kecepatan sudut, serta nilai persentase *throttle* dari *remote control*. Nilai konstanta *thrust* dan konstanta *drag,* dilakukan pengujian pada masing-masing motor dengan diberi suplai sinyal PWM, lalu diukur kecepatan dari keempat motor tersebut, dan dihitung konstanta *thrust* dan *drag* dari *quadcopter*. Parameter-parameter *quadcopter* diambil dari penelitian sebelumnya dapat dilihat pada Tabel 3.1[12].

**Tabel 3.1** Parameter *Quadcopter*

| No. | Parameter | Nilai | Satuan |
|---|---|---|---|
| 1, | Massa *quadcopter* | 1.26 | Kg |
| 2. | Jari-jari *quadcopter* | 0,206 | Meter |
| 3. | Momen inersia sumbu X | $1{,}68 \times 10^{-3}$ | $Kg.m^2$ |
| 4. | Momen inersia sumbu Y | $1{,}68 \times 10^{-3}$ | $Kg.m^2$ |
| 5. | Momen inersia sumbu Z | $1{,}25 \times 10^{-3}$ | $Kg.m^2$ |

| No. | Parameter | Nilai | Satuan |
|---|---|---|---|
| 6. | Konstanta drag | 4,19 x $10^{-5}$ | Nm.sec$^2$ |
| 7. | Konstanta Thrust | 1,68918 x $10^{-6}$ | N.sec$^2$ |
| 8. | Gaya Grafitasi | 9.8 | m/sec$^2$ |

Pada bab 2 telah diperoleh model matematik dari *quadcopter*, parameter-parameter *quadcopter* yang telah diketahui disubtistusikan ke dalam model matematika *quadcopter* sehingga model matematika bisa digunakan pada simulasi. Model matematika *quadcopter* dapat dituliskan kembali menjadi Persamaan (3.31).

$$\begin{cases} \ddot{X} = \frac{U_1}{1.26}(cos\psi sin\theta cos\phi + sin\psi sin\phi) \\ \ddot{Y} = \frac{U_1}{1.26}(sin\psi sin\theta cos\phi - cos\psi sin\phi) \\ \ddot{Z} = -9.81 + (cos\theta cos\phi)\frac{U_1}{1.26} \\ \dot{p} = -0.5495qr - 0.00017q\Omega + 0.2052U_2 \\ \dot{q} = 0{,}1675\text{pr} - 0{,}0094p\Omega + 2{,}955U_3 \\ \dot{r} = -2{,}0257pq + 0.0954U_4 \end{cases} \quad (3.31)$$

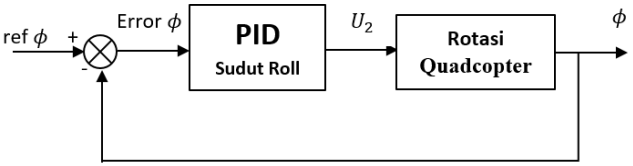
Dari identifikasi konstanta dilakukan pembatasan, di antaranya sudut *roll* dan *pitch* adalah maksimum 0,7 rad. Secara analisis sederhana diperoleh bahwa respon *output* Z , sudut *roll*, sudut *pitch*, dan sudut *yaw* terhadap *input* $U_1$, $U_2$, $U_3$,dan $U_4$.

## 3.3 Perancangan Kontroler PID

Kontroler PID digunakan untuk mengendalikan gerak rotasi pada *quadcopter* yang meliputi sudut roll, sudut *pitch* dan sudut *yaw*. Pengendalian sudut pada *quadcopter* dilakukan untuk menjaga arah dari quadcopter agar tetap. Perancangan kontroler PID untuk gerak rotasi diperlukan suatu linierisasi pada sistem *quadcopter* dikarenakan sistem pada *quadcopter* merupakan sistem nonlinier. Metode linierisasi yang akan digunakan adalah *feedback linierization,* dengan memodifikasi sinyal kontrol untuk menghilangkan efek nonlinier dari sistem. Model yang dipakai pada pengaturan sudut adalah Persamaan orde satu dengan *gain* overall 1 dan konstanta waktu 0,5 detik. Diharapkan sudut dapat merespon sangat cepat karena akan dijadikan variabel sinyal kontrol pada pengaturan gerak translasi atau paling tidak secara kaidah hukum pengaturan cascade bahwa loop dalam harus lebih cepat dari loop luar.

### 3.3.1 PID *Roll*

Perancangan kontroler PID untuk sudut *roll quadcopter* dtunjukkan pada diagram blok Gambar 3.3. Sudut *roll* merupakan sudut yang menyebabkan *quadcopter* bergerak pada sumbu Y. Pada diagram blok Gambar 3.3, masukan sistem berupa sudut *roll* referensi yang dihasilkan oleh sinyal kontrol gerak translasi sumbu Y. Pengendalian sudut *roll* dapat dilakukan dengan mengendalikan sinyal kontrol $U_2$.




**Gambar 3.3** Diagram Blok Perancangan Kontroler PID Sudut *Roll*

Perancangan kontroler PID dimulai dari linierisasi *feedback* untuk sudut *roll*, sebagai berikut :

$$\dot{\phi} = p \tag{3.2}$$

$$\ddot{\phi} = \dot{p} \tag{3.3}$$

$$\ddot{\phi} = -0{,}5495qr - 0{,}00017q\Omega + 0{,}2052U_2 \tag{3.4}$$

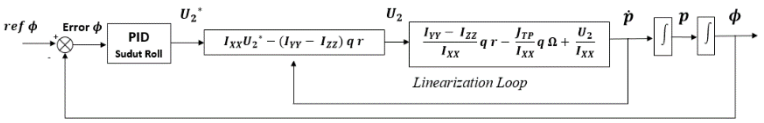
Persamaan (3.4) dimodifikasidan dibuat permisalan pada Persamaan (3.5) sehingga didapatkan Persamaan (3.6) dan persamaan sudut roll menjadi Persamaan (3.7).menjadi Persamaan (3.6) dan Persamaan (3.7)

$$\ddot{\phi} = U_2{}^* \tag{3.5}$$

$$U_2 = 2{,}677qr + 0.00082q\Omega + 4{,}873U_2{}^* \tag{3.6}$$

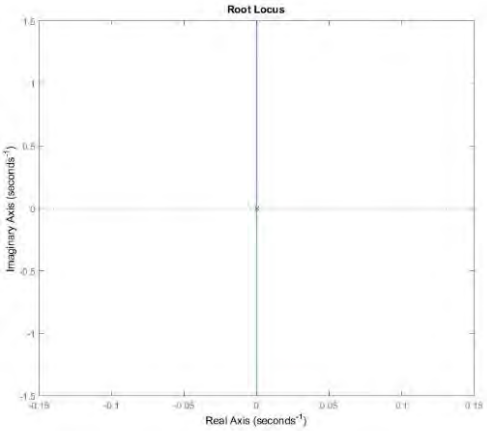
$$\phi = \frac{1}{s}\, p \tag{3.7}$$

Diagram blok *feedback* linearization untuk sudut *roll* dapat dilihat pada Gambar 3.4.




**Gambar 3.4** Diagram Blok *Feedback* Linearization untuk Sudut *Roll*

Dari Persamaa (3.7) maka dapat disimpukan model sistem merupakan sistem *double integrator.* Bila menggunakan kontroler PID, maka diperlukan metode tunning untuk mendapatkan parameter dari kontroler PID tersebut. Metode tunning yang paling terkenal adalah metode *Ziegler-Nichols*. Namun metode *Ziegler-Nichol*s konvensional tidak dapat diterapkan karena respon *closed loop* dari sistem selalu berosilasi. Hal ini dapat dibuktikan dengan menggunakan analisa kestabilan root locus untuk *closed loop* kontroler proporsional dan *plant*. Dari analisis *root locus* pada Gambar 3.4 diperlihatkan bahwa untuk setiap K, sistem akan berada pada daerah *critically stable*. Bila didekatkan dengan prosedur *Ziegler-Nichols*, dapat ditarik kesimpulan, bahwa metode *Ziegler-Nichols* "konvensional" tidak dapat diterapkan untuk mencari nilai parameter kontroler PID.




**Gambar 3.5** Grafik Root Locus Persamaan Sudut *Roll*

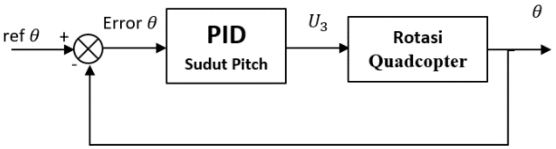
Sehingga pada tugas akhir kali ini menggunakan metode tuning manual, sebagai berikut :

1. Langkah awal adalah menambah *gain* kontroler proporsional hingga keadaan stabil namun respon masih ada isolasidan mengabaikan *gain* kontroler integral dan deferensial.
2. Untuk meredam osilasi pada respon, tambahkan *gain* kontroler deferensial sampai respon lebih stabil.
3. Selanjutnya tambahkan nilai *gain* kontroler integral untuk menambah kestabilan pada respon.

Sehingga parameter PID didapatkan Kp sebesar 800, Ki sebesar 0,01, dan Kd sebesar 2800.

### 3.3.2 PID *Pitch*

Perancangan kontroler PID untuk sudut *pitch quadcopter* dtunjukkan pada diagram blok Gambar 3.6. Sudut *pitch* merupakan sudut yang menyebabkan *quadcopter* bergerak pada sumbu X. Pada diagram blok Gambar 3.6, masukan sistem berupa sudut *pitch* referensi yang dihasilkan oleh sinyal kontrol gerak translasi sumbu X. Pengendalian sudut *pitch* dapat dilakukan dengan mengendalikan sinyal kontrol $U_3$.




**Gambar 3.6** Diagram Blok Perancangan Kontroler PID Sudut *Pitch*

Perancangan kontroler PID dimulai dari linierisasi *feedback* untuk sudut *pitch*, didapatkan persamaan sebagai berikut :

$$\dot{\theta} = q \tag{3.8}$$

$$\ddot{\theta} = \dot{q} \tag{3.9}$$

$$\ddot{\theta} = 0{,}1675\mathrm{pr} - 0{,}0094p\Omega + 2{,}955U_3 \tag{3.10}$$

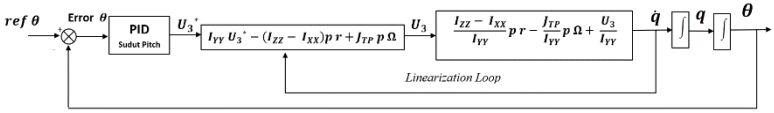
Persamaan (3.10) dimodifikasidan dibuat permisalan pada Persamaan (3.11) sehingga didapatkan Persamaan (3.12) dan persamaan sudut pitch menjadi Persamaan (3.13).

$$\ddot{\theta} = U_3^{*} \tag{3.11}$$

$$U_3 = -0{,}056\text{pr} + 0{,}0031p\Omega + 0{,}3384{U_3}^* \quad (3.12)$$

$$\theta = \frac{1}{s}\, q \quad (3.13)$$

Diagram blok *feedback* linearization untuk sudut *roll* dapat dilihat pada Gambar 3.7.

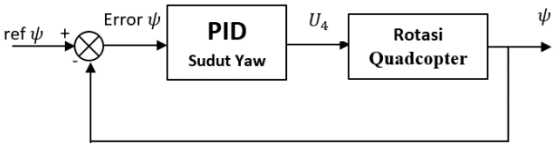



**Gambar 3.7** Diagram Blok *Feedback* Linearization untuk Sudut *Pitch*

Dari Persamaa (3.12) dan Persamaan (3.13) maka dapat disimpukan model sistem merupakan sistem *double integrator*. Sama seperti sudut *Roll*, parameter kontroler PID untuk sudut *Pitch* menggunakan metode tuning manual. Sehingga parameter PID didapatkan nilai Kp sebersar 400, Kd sebesar 2250, dan Ki sebesar 0,01.

### 3.3.3 PID *Yaw*

Perancangan kontroler PID untuk sudut *yaw quadcopter* dtunjukkan pada diagram blok Gambar 3.8. Sudut *pitch* merupakan sudut yang menyebabkan *quadcopter* bergerak memutar pada poros sumbu Z.




**Gambar 3.8** Diagram Blok Perancangan Kontroler PID Sudut *Yaw*

Perancangan kontroler PID dimulai dari linierisasi *feedback* untuk sudut *yaw*, sebagai berikut :

$$\dot{\psi} = r \quad (3.14)$$

$$\ddot{\psi} = \dot{r} \quad (3.15)$$

$$\ddot{\psi} = -2{,}0257pq + 0{,}0954U_4 \quad (3.16)$$

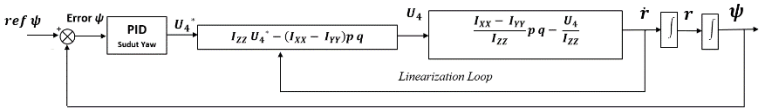
Persamaan (3.16) dimodifikasidan dibuat permisalan pada Persamaan (3.17) sehingga didapatkan Persamaan (3.18) dan persamaan sudut yaw dapat ditulis menjadi Persamaan (3.19).

$$\ddot{\psi} = U_4^* \tag{3.17}$$

$$U_4 = 21{,}23pq + 10{,}482\, U_4^* \tag{3.18}$$

$$\psi = \frac{1}{s}\, r \tag{3.19}$$

Diagram blok *feedback* linearization untuk sudut *yaw* dapat dilihat pada Gambar 3.9.




**Gambar 3.9** Diagram Blok *Feedback* Linearization untuk Sudut *Pitch*

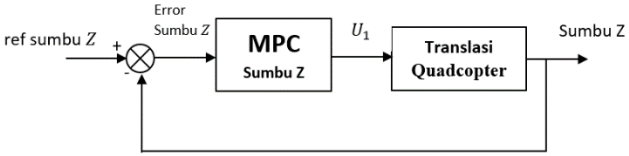
Dari Persamaa (3.18) dan Persamaan (3.19) maka dapat disimpukan model sistem merupakan sistem *double integrator.* Dari Persamaa (3.19) maka dapat disimpukan model sistem merupakan sistem *double integrator.* Sama seperti sudut *Roll* dan *Pitch*, parameter kontroler PID untuk sudut *Yaw* menggunakan metode tuning manual. Sehingga parameter PID didapatkan nilai Kp sebersar 300, Kd sebesar 1400, dan Ki sebesar 0,0001.

## 3.4 Perancangan Kontroler MPC

Perancangan kontroler MPC ini dimaksudkan untuk mengontrol gerak *quadcopter* pada sumbu X,Y dan Z. Tahapan desain kontroler ini meliputi diskritisasi sistem dari persamaan linier yang tetah didapat, desain augmented model dari *state space* yang sudah didapatkan dari proses diskritisasi, perancangan *state estimator* pada sistem menggunakan *observer* dan penentuan parameter kontroler *Model Predictive Control* (MPC).

### 3.4.1 Kontroler MPC pada Sumbu Z

Perlu diketahui bahwa gerakan throttle pada sumbu Z tidak memerlukan gerakan rotasi seperti pada gerak untuk sumbu X maupun Y. Perancangan kontroler MPC pada sumbu Z dapat dilihat pada blok diagram Gambar 3.10.




**Gambar 3.10** Diagram Blok Perancangan Kontroler MPC Sumbu Z

***a. Discretization***

Persamaan linier untuk sumbu Z sebagai berikut :

$$\ddot{Z} = -9.81 + (cos\theta cos\phi)\frac{U_1}{1.26} \tag{3.20}$$

Untuk merancang kontroler MPC, sistem yang akan dikontrol harus berbentuk model *state space,* sehingga Persamaan (3.20) akan dimodifikasi agar dapat dibuat model *state space*. Modifikasi persamaan linier sumbu Z dapat dilihat pada Persamaan (3.21).

$$U_z = -9.81 + (cos\theta cos\phi)\frac{U_1}{1.26} \tag{3.21}$$

Jika Persamaan (3.21) disubtitusi ke dalam Persamaan (3.20) maka menjadi Persamaan (3.22).

$$\ddot{Z} = U_z \tag{3.22}$$

Sehingga dari Persamaan (3.22) dapat dibuat model *state space* pada Persamaan (3.23).

$$\begin{bmatrix}\dot{Z}\\ \ddot{Z}\end{bmatrix} = \begin{bmatrix}0 & 1\\ 0 & 0\end{bmatrix}\begin{bmatrix}Z\\ \dot{Z}\end{bmatrix} + \begin{bmatrix}0\\ 1\end{bmatrix}U_z \tag{3.23}$$

Dari *state space* Persamaan (3.23) didapatkan matriks A dan B pada Persamaan (3.24).

$$A = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \quad , B = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \tag{3.24}$$

Pemilihan matriks C tergantung pada state mana yang akan dihitung. Pada tugas akhir ini, posisi yang akan dihitung nilainya, sehingga dapat menggunakan output matriks $C = [1 \quad 0]$. Selanjutnya adalah diskritisasi Persamaan (3.23) dengan menggunakan metode forward euler sebagai berikut :

$$\begin{bmatrix} Z(k+1) \\ V_z(k+1) \end{bmatrix} = \begin{bmatrix} 1 & \Delta T \\ 0 & 1 \end{bmatrix} \begin{bmatrix} Z(k) \\ V_z(k) \end{bmatrix} + \begin{bmatrix} 0 \\ \Delta T \end{bmatrix} U_z(k) \tag{3.25}$$

$$Y_z(k) = [1 \quad 0] \begin{bmatrix} Z(k) \\ V_z(k) \end{bmatrix} \tag{3.26}$$

Dimana $\Delta T$ merupakan *sampling interval* dan dipilih nilai $\Delta T$ sebersar 0.1s. State $Z(k)$ dan $V_z(k)$ merepresentasikan posisi dan kecepatan.

**b. Desain *Augmented Model* untuk Sumbu Z**

Desain augmented model adalah dengan cara mengubah bentuk *state space* sumbu Z ke dalam bentuk augmented model dari model yang sudah didiskritisasi sebagai berikut :

$$\begin{bmatrix} Z(k+1) \\ V_z(k+1) \end{bmatrix} = \overbrace{\begin{bmatrix} 1 & 0.01 \\ 0 & 1 \end{bmatrix}}^{A_m} \begin{bmatrix} Z(k) \\ V_z(k) \end{bmatrix} + \overbrace{\begin{bmatrix} 0 \\ 0.01 \end{bmatrix}}^{B_m} U_z(k) \tag{3.27}$$

$$Y_z(k) = \overbrace{[1 \quad 0]}^{C_m} \begin{bmatrix} Z(k) \\ V_z(k) \end{bmatrix} \tag{3.28}$$

Berdasarkan persamaan (2.65) , Persamaan (3.27) dan Persamaan (3.28) dapat diubah menjadi bentuk *augmented model* pada Persamaan (3.29). Matriks A,B, dan C yang berupa augmented model akan digunakan selanjutnya dalam merancang kontroler MPC.

$$\overbrace{\begin{bmatrix}\Delta Z_m(k+1)\\ Y_z(k+1)\end{bmatrix}}^{x(k+1)} = \overbrace{\begin{bmatrix}1 & 0{,}01 & 0\\ 0 & 1 & 0\\ 1 & 0{,}01 & 1\end{bmatrix}}^{A}\begin{bmatrix}\Delta Z_m\\ Y_z(k)\end{bmatrix} + \overbrace{\begin{bmatrix}0{,}0001\\ 0{,}01\\ 0{,}0001\end{bmatrix}}^{B}\Delta U_z(k) \quad (3.29)$$

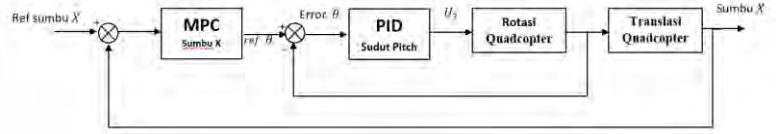
$$Y_z(k) = \overbrace{\begin{bmatrix}1 & 0 & 0\end{bmatrix}}^{C}\begin{bmatrix}\Delta Z_m\\ Y_z(k)\end{bmatrix} \quad (3.31)$$

**c. Penentuan *Gain* dan Prameter Kontroler MPC untuk Sumbu Z**

Langkah selanjutnya adalah menentukan parameter kontroler MPC. Pada perancangan kontroler MPC untuk sumbu Z dipilih *parameter prediction horizon* sebesar 10 langkah, *control horizon* sebesar 3 dan *tuning parameter* indeks performansi sebesar 1. Selanjutnya adalah mencari nilai *gain* kontroler MPC ($K_{mpc}$dan $K_y$ ). Dimana *gain* $K_y$ didapat dari baris pertama dari matriks $(\Phi^T\Phi + \bar{R}\,)^{-1}\Phi^T R_s$ dan *gain* $K_{mpc}$ didapat dari baris pertama dari matriks $(\Phi^T\Phi + \bar{R}\,)^{-1}\Phi^T F$. Sehingga nilai *gain* $K_{y\_z} = 0{,}83$ dan nilai *gain* $K_{mpc\_z}$ =[6.90,3.23,0.878]

### 3.4.2 Kontroler MPC pada Sumbu X

Gerakan translasi sumbu X dipengaruhi oleh gerakan rotasi sudut *pitch*, sistem yang digunakan menggunakan sistem casecade yang dapat dlihat pada blok diagram Gambar 3.11.




**Gambar 3.11** Diagram Blok Perancangan Kontroler MPC Sumbu X

***a. Discretization***

Persamaan (3.35) memiliki tiga *variable* $\phi, \theta, \psi$ dimana gerak pada sumbu $X$ bergantung pada besarnya nilai sudut $\theta$ dan juga mendapat

pengaruh secara langsung oleh sudut $\phi$ . Dimisalkan bahwa *quadcopter* bersifat *rigid* untuk sudut $\psi$ sehingga besarnya perubahan sudut *yaw* bernilai sangat kecil atau $\psi \approx 0$ . Sehingga persamaan linier untuk sumbu X menjadi Persamaan (3.36). Persamaan linier untuk sumbu X sebagai berikut :

$$\ddot{X} = \frac{U_1}{1.26}(cos\psi sin\theta cos\phi + sin\psi sin\phi) \tag{3.35}$$

$$\ddot{X} = \frac{U_1}{1.26}(sin\theta cos\phi) \tag{3.36}$$

Untuk merancang kontroler MPC, sistem yang akan dikontrol harus berbentuk model *state space,* sehingga Persamaan (3.36) akan dimodifikasi agar dapat dibuat model *state space*. Modifikasi persamaan linier sumbu X dapat dilihat pada Persamaan (3.37).

$$U_x = \frac{U_1}{1.26}(sin\theta cos\phi) \tag{3.37}$$

Jika Persamaan (3.37) disubtitusi ke dalam Persamaan (3.36) maka menjadi Persamaan (3.38).

$$\ddot{X} = U_x \tag{3.38}$$

Sehingga dari Persamaan (3.38) dapat dibuat model *state space* pada Persamaan (3.39).

$$\begin{bmatrix} \dot{X} \\ \ddot{X} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} X \\ \dot{X} \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} U_x \tag{3.39}$$

Dari *state space* Persamaan (3.39) didapatkan matriks A dan B pada Persamaan (3.40).

$$A = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \quad , B = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \tag{3.40}$$

Pemilihan matriks C tergantung pada state mana yang akan dihitung. Pada tugas akhir ini, posisi yang akan dihitung nilainya,

sehingga dapat menggunakan output matriks $C = [1 \quad 0]$. Selanjutnya adalah diskritisasi Persamaan (3.39) dengan menggunakan metode forward euler seperti pada Persamaan (3.25) dan Persamaan (3.26). *Sampling interval* ($\Delta T$) yang dipilih nilai $\Delta T$ sebersar 0.1s. State $X(k)$ dan $V_x(k)$ merepresentasikan posisi dan kecepatan.

**b. Desain *Augmented Model* untuk Sumbu X**

Desain augmented model adalah dengan cara mengubah bentuk *state space* sumbu X ke dalam bentuk augmented model dari model yang sudah didiskritisasi sebagai berikut :

$$\begin{bmatrix} X(k+1) \\ V_x(k+1) \end{bmatrix} = \overbrace{\begin{bmatrix} 1 & 0.01 \\ 0 & 1 \end{bmatrix}}^{A_m} \begin{bmatrix} X(k) \\ V_x(k) \end{bmatrix} + \overbrace{\begin{bmatrix} 0 \\ 0.01 \end{bmatrix}}^{B_m} U_x(k) \tag{3.41}$$

$$Y_x(k) = \overbrace{[1 \quad 0]}^{C_m} \begin{bmatrix} X(k) \\ V_x(k) \end{bmatrix} \tag{3.42}$$

Berdasarkan persamaan (2.65) , Persamaan (3.41) dapat diubah menjadi bentuk *augmented model* pada Persamaan (3.43). Matriks A,B, dan C yang berupa augmented model akan digunakan selanjutnya dalam merancang kontroler MPC.

$$\overbrace{\begin{bmatrix} \Delta X_m(k+1) \\ Y_x(k+1) \end{bmatrix}}^{x(k+1)} = \overbrace{\begin{bmatrix} 1 & 0{,}01 & 0 \\ 0 & 1 & 0 \\ 1 & 0{,}01 & 1 \end{bmatrix}}^{A} \begin{bmatrix} \Delta X_m \\ Y_x(k) \end{bmatrix} + \overbrace{\begin{bmatrix} 0{,}0001 \\ 0{,}001 \\ 0{,}0001 \end{bmatrix}}^{B} \Delta U_x(k) \tag{3.43}$$

$$Y_x(k) = \overbrace{[1 \quad 0 \quad 0]}^{C} \begin{bmatrix} \Delta X_m \\ Y_x(k) \end{bmatrix} \tag{3.44}$$

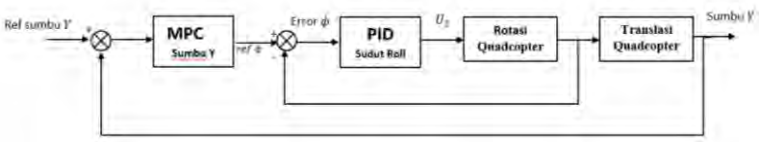
**c. Penentuan *Gain* dan Prameter Kontroler MPC untuk Sumbu X**

Langkah selanjutnya adalah menentukan parameter kontroler MPC. Pada perancangan kontroler MPC untuk sumbu X dipilih parameter prediction *horizon* sebesar 20 langkah, *control horizon* sebesar 2 dan tuning parameter indeks performansi sebesar 1. Selanjutnya adalah

mencari nilai *Gain* kontroler MPC ($K_{mpc}$dan $K_y$ ). Dimana *gain* $K_y$ didapat dari baris pertama dari matriks $(\Phi^T\Phi + \bar{R}\,)^{-1}\Phi^T R_s$ dan *gain* $K_{mpc}$ didapat dari baris pertama dari matriks $(\Phi^T\Phi + \bar{R}\,)^{-1}\Phi^T F$. Sehingga nilai *gain* $K_{y\_x} = 0{,}14$, dan nilai *gain* $K_{mpc\_x} = [2.197{,}0.191{,}0.143]$.

### 3.4.3 Kontroler MPC pada Sumbu Y

Gerakan translasi sumbu Y dipengaruhi oleh gerakan rotasi sudut *roll*, sistem yang digunakan menggunakan sistem casecade yang dapat dlihat pada blok diagram Gambar 3.12.




**Gambar 3.12** Diagram Blok Perancangan Kontroler MPC Sumbu Y

***a. Discretization***

Persamaan linier untuk sumbu Y sebagai berikut :

$$\ddot{Y} = \frac{U_1}{1.26}(sin\psi sin\theta cos\phi - cos\psi sin\phi) \tag{3.45}$$

Persamaan tersebut memiliki tiga variabel $\phi, \theta, \psi$ dimana gerak pada sumbu Y bergantung pada besarnya nilai sudut $\phi$ dan juga mendapat pengaruh secara langsung oleh sudut $\theta$. Dimisalkan bahwa *quadcopter* bersifat *rigid* untuk sudut $\psi$ sehingga besarnya perubahan sudut *yaw* bernilai sangat kecil atau $\psi \approx 0$. Sehingga persamaan linier untuk sumbu Y menjadi Persamaan (3.46).

$$\ddot{Y} = -(\sin\phi)\frac{U_1}{1.26} \tag{3.46}$$

Untuk merancang kontroler MPC, sistem yang akan dikontrol harus berbentuk model *state space,* sehingga Persamaan (3.46) akan dimodifikasi agar dapat dibuat model *state space*. Modifikasi persamaan linier sumbu Y dapat dilihat pada Persamaan (3.47).

$$U_y = -(\sin\phi)\frac{U_1}{1.26} \tag{3.47}$$

Jika Persamaan (3.47) disubtitusi ke dalam Persamaan (3.46) maka menjadi Persamaan (3.48).

$$\ddot{Y} = U_y \tag{3.48}$$

Sehingga dari Persamaan (3.48) dapat dibuat model *state space* pada Persamaan (3.49).

$$\begin{bmatrix}\dot{Y}\\ \ddot{Y}\end{bmatrix} = \begin{bmatrix}0 & 1\\ 0 & 0\end{bmatrix}\begin{bmatrix}Y\\ \dot{Y}\end{bmatrix} + \begin{bmatrix}0\\ 1\end{bmatrix}U_y \tag{3.49}$$

Dari *state space* Persamaan (3.49) didapatkan matriks A dan B pada Persamaan (3.50).

$$A = \begin{bmatrix}0 & 1\\ 0 & 0\end{bmatrix} \quad , B = \begin{bmatrix}0\\ 1\end{bmatrix} \tag{3.50}$$

Pemilihan matriks $C$ tergantung pada state mana yang akan dihitung. Pada tugas akhir ini, posisi yang akan dihitung nilainya, sehingga dapat menggunakan output matriks $C = [1 \quad 0]$. Selanjutnya adalah diskritisasi Persamaan (3.49) dengan menggunakan metode forward euler seperti pada Persamaan (3.25) dan Persamaan (3.26). *Sampling interval* ($\Delta T$) yang dipilih nilai $\Delta T$ sebersar 0.01s. State $Y(k)$ dan $V_y(k)$ merepresentasikan posisi dan kecepatan.

**b. Desain *Augmented Model* untuk Sumbu Y**

Desain augmented model adalah dengan cara mengubah bentuk *state space* sumbu Y ke dalam bentuk augmented model dari model yang sudah didiskritisasi sebagai berikut :

$$\begin{bmatrix}Y(k+1)\\ V_y(k+1)\end{bmatrix} = \overbrace{\begin{bmatrix}1 & 0.01\\ 0 & 1\end{bmatrix}}^{A_m}\begin{bmatrix}Y(k)\\ V_y(k)\end{bmatrix} + \overbrace{\begin{bmatrix}0\\ 0.01\end{bmatrix}}^{B_m}U_y(k) \tag{3.51}$$

$$Y_y(k) = \overbrace{\begin{bmatrix}1 & 0\end{bmatrix}}^{C_m}\begin{bmatrix}Y(k)\\ V_y(k)\end{bmatrix} \tag{3.52}$$

Berdasarkan persamaan (2.65) , Persamaan (3.51) dapat diubah menjadi bentuk *augmented model* pada Persamaan (3.53).
Matriks A,B, dan C yang berupa augmented model akan digunakan selanjutnya dalam merancang kontroler MPC.

$$\overbrace{\begin{bmatrix}\Delta Y_m(k+1)\\ Y_y(k+1)\end{bmatrix}}^{x(k+1)} = \overbrace{\begin{bmatrix}1 & 0{,}01 & 0\\ 0 & 1 & 0\\ 1 & 0{,}01 & 1\end{bmatrix}}^{A}\begin{bmatrix}\Delta Y_m\\ Y_y(k)\end{bmatrix} + \overbrace{\begin{bmatrix}0{,}0001\\ 0{,}001\\ 0{,}0001\end{bmatrix}}^{B}\Delta U_y(k) \tag{3.53}$$

$$Y_y(k) = \overbrace{\begin{bmatrix}1 & 0 & 0\end{bmatrix}}^{C}\begin{bmatrix}\Delta Y_m\\ Y_y(k)\end{bmatrix} \tag{3.54}$$

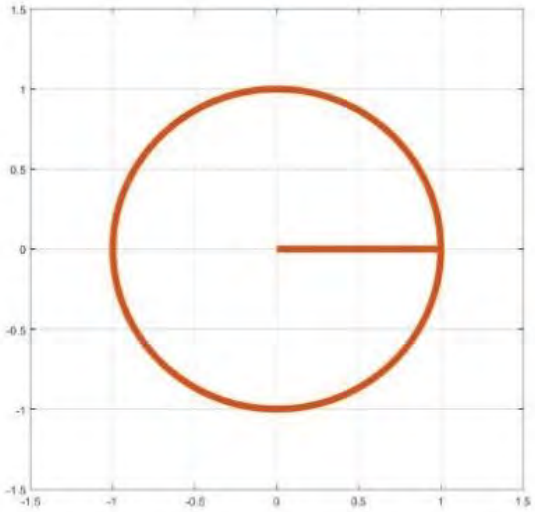
**c. Penentuan *Gain* dan Prameter Kontroler MPC untuk Sumbu Y**

Langkah selanjutnya adalah menentukan parameter kontroler MPC. Pada perancangan kontroler MPC untuk sumbu Y dipilih parameter prediction *horizon* sebesar 14 langkah, *control horizon* sebesar 2 dan tuning parameter indeks performansi sebesar 1. Selanjutnya adalah mencari nilai *Gain* kontroler MPC ($K_{mpc}$dan $K_y$ ). Didapatkan nilai *gain* $K_{y\_y}$ =0,6444, dan nilai *gain* $K_{mpc\_y} = [0.55, 0.034, 0.05]$.

## 3.5 Perancangan *Trajectory Tracking*

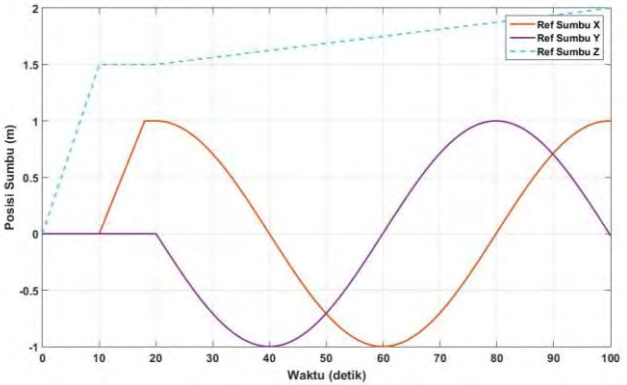
Referensi jalur yang akan disimulasikan untuk *Trajectory Tracking quadcopter* berbentuk lingkaran ditunjukkan pada Gambar 3.13 dan berbentuk segiempat ditunjukkan pada Gambar 3.15. Lintasan lingkaran dipilih untuk menguji kemampuan *quadcopter* dalam menjejak lintasan dengan referensi yang terus berubah. Perubahan referensi membuat kecepatan *quadcopter* berubah pula. Lintasan tersebut juga membuat *quadcopter* melakukan gerakan maneuver membentuk lingkaran.

Sehingga, dapat diketahui respon *quadcopter* saat melakukan manuver gerakan *cruise*. Lintasan lingkaran dengan jari-jari 1 m, kecepatan sudut 0,1 rad/s, dan waktu keseluruhan simulasi adalah 100 detik. Untuk referensi sumbu Z dibuat tetap sebesar 1.5 m sampai detik ke 20, lalu bergerak naik sampai nilai 2 m, sedangkan referensi sumbu X dan Y berubah-ubah berbentuk sinyal sinusoidal yang ditunjukkan pada Gambar 3.11.



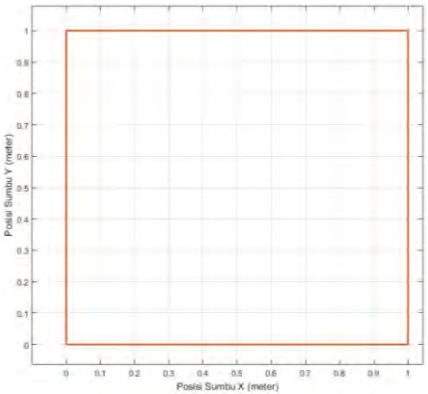

**Gambar 3.13** Referensi *Trajectory Tracking* Lingkaran *Quadcopter* Sumbu X dan Y

*Quadcopter* melakukan take off vertical sampai ketinggian 2 meter setelah itu *quadcopter* bergerak ke koordinat (1,0,2) kearah sumbu x selama 8 detik. *quadcopter* melakukan hover pada koordinat (1,0,2) selama 2 detik kemudian *quadcopter* mulai menjejak lintasan yang sudah disianpak.

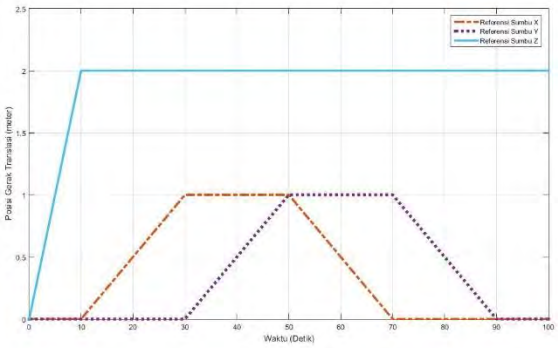



**Gambar 3.14** Grafik Referensi Sumbu X, Y dan Z Terhadap Waktu

Sedangkan untuk lintasan segiempat dengan panjang 1 meter dan lebar 1 meter. Referensi sumbu Z dibuat tetap sebesar 2 m sedangkan referensi sumbu X dan Y berubah-ubah membentuk sinyal ramp ditunjukkan pada Gambar 3.16 . Waktu simulasi untuk lintasan segiempat adalah 100 detik. referensi segiempat dipilih untuk mengetahui respon *quadcopter* saat menjejak lintasan bila referensi cenderung tetap.




**Gambar 3.15** Referensi *Trajectory Tracking* Segiempat *Quadcopter* Sumbu X dan Y




**Gambar 3.16** Referensi Sumbu X, Y dan Z Lintasan Segiempat

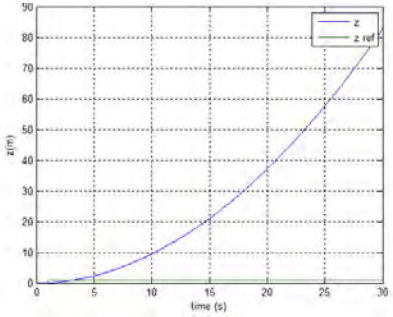
*[Halaman ini sengaja dikosongkan]*

# BAB IV
# ANALISIS DATA DAN PEMBAHASAN

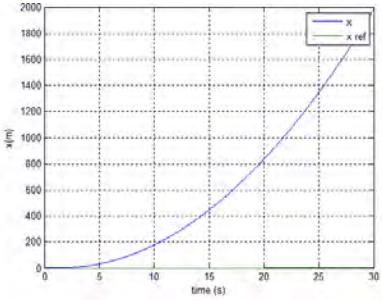
Bab ini berisi tentang hasil simulasi dan implementasi serta *analisis* dari perancangan yang sudah dilakukan pada Bab III. Pengujian akan dilakukan dengan dua kontroler untuk *inner loo*p dan *outer loop* sistem. Pengujian kontroler juga dengan memberikan gangguan pada sistem.

## 4.1 Simulasi *Open Loop* Sistem

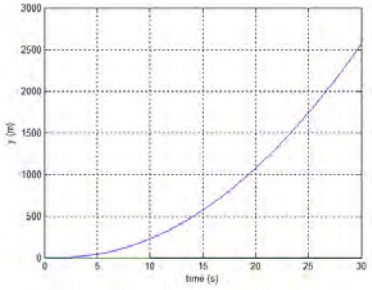
Simulasi *open loop* sistem dilakukan untuk mengetahui karakteristik dari *quadcopter* sebelum diberikan kontroler. Pengujian dilakukan dengan memberikan nilai akhir sinyal step 1 pada sumbu X, Y dan Z. Hasil respon sumbu X dapat dilihat pada Gambar 4.2, hasil respon untuk sumbu Y pada Gambar 4.3, sedangkan hasil respon sumbu Z pada Gambar 4.1.




**Gambar 4.1** Respon *Open Loop* Posisi Z tanpa Kontroler




**Gambar 4.2** Respon *Open Loop* Posisi X tanpa Kontroler




**Gambar 4.3** Respon *Open Loop* Posisi Y tanpa Kontroler

## 4.2 Simulasi Pengujian dengan Kontroler PID

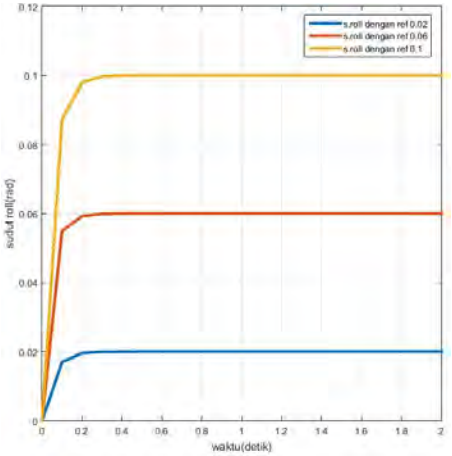
Simulasi pengujian dengan kontroler PID pada gerak rotasi terdiri dari 2 cara yaitu pengujian dengan variasi set point/nilai referensi dan pengujian dengan variasi *initial condition*/nilai awal. Pada simulasi tugas akhir ini hanya menggunakan gerak rotasi sudut *roll* dan sudut *pitch* dikarenakan sudut *yaw* yang dianggap rigid dan nilai perubahan sudut dibatasi bernilai maksimal 0,1 dikarenakan sudut rotasi *quadcopter* harus bernilai 0 untuk menjaga kestabilan *quadcopter*. Pengujian dengan variasi set point dilakukan untuk mengetahui apakah respon kontroler PID pada gerak rotasi sudut *roll* maupun sudut *pitch* dapat mengikuti referensi yang diberikan. Sedangkan pengujian variasi nilai awal dilakukan untuk mengetahui apakah respon kontroler PID pada gerak rotasi dapat meregulasi ke titik awal, sehingga apabila *quadcopter* diberi gangguan, maka akan tetap kembali ke jalur yang ditentukan.

### 4.2.1 Simulasi Pengujian dengan Kontroler PID Sudut *Roll*

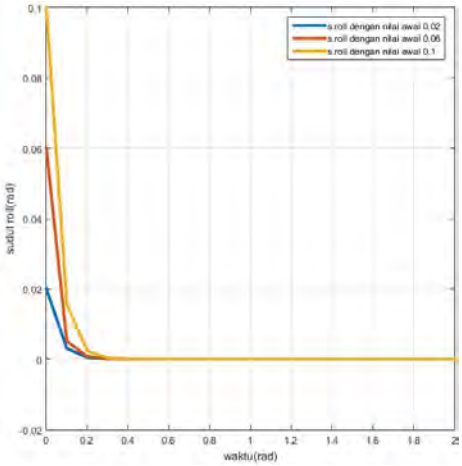
Pada simulasi pengujian dengan kontroler PID sudut *roll* diberikan variasi set point sebesar 0,02 radian, 0,06 radian dan 0,1 radian dengan nilai awal 0 radian terlebih dahulu, set point yang diberikan berupa sinyal step. Hasil pengujian set point untuk sudut *roll* dapat dilihat pada Gambar 4.4. Didapatkan bahwa respon degan set point sebesar 0,02 radian, 0,06 radian dan 0,1 radian mengikuti referensi yang diberikan.

Selanjutnya pengujian sudut *roll* dengan variasi nilai awal. Pada pengujian ini diberikan nilai awal sebesar sebesar 0,02 radian, 0,06 radian dan 0,1 radian dengan nilai set point 0 radian. Hasil respon dapat dilihat

pada Gambar 4.5, didapatkan bahwa pada semua nilai awal, respon tetap meregulasi ke titik awal/ 0 radian.




**Gambar 4.4** Respon Simulasi Sudut *Roll* dengan Variasi Nilai Referensi



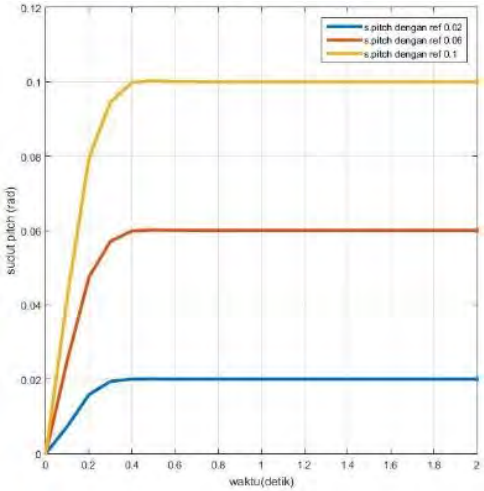

**Gambar 4.5** Respon Simulasi Sudut *Roll* dengan Variasi Nilai Awal

### 4.2.2 Simulasi Pengujian dengan Kontroler PID Sudut *Pitch*

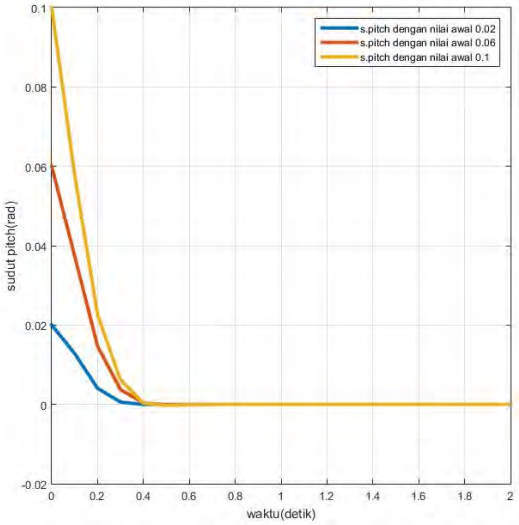
Pada simulasi pengujian dengan kontroler PID sudut *pitch* sama dengan simulasi pengujian sudut *roll* yang diberikan variasi set point maupun dengan variasi nilai awal.

Pengujian pertama adalah pengujian dengan variasi set point sebesar 0,02 radian, 0,06 radian dan 0,1 radian dengan nilai awal 0 radian dan set point berupa sinyal step. Hasil pengujian dapat dilihat pada Gambar 4.6, didapatkan bahwa pada semua nilai set point, respon tetap mengikuti referensi yang diberikan.

Selanjutnya pengujian sudut *pitch* dengan variasi nilai awal. Pada pengujian ini diberikan nilai awal sebesar sebesar 0,02 radian, 0,06 radian dan 0,1 radian dengan nilai set point 0 radian Hasil respon dapat dilihat pada Gambar 4.7, didapatkan bahwa pada semua nilai awal, respon tetap meregulasi ke titik awal/ 0 radian.




**Gambar 4.6** Respon Simulasi Sudut *Pitch* dengan Variasi Nilai Awal




**Gambar 4.7** Respon Simulasi Sudut *Pitch* dengan Variasi Nilai Referensi

## 4.3 Simulasi Pengujian Kontroler MPC

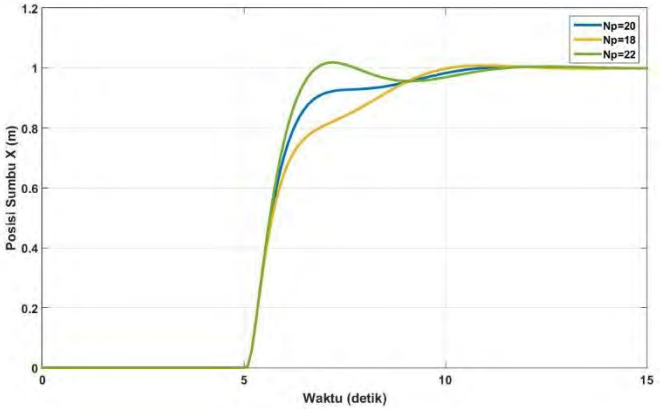
Pada penelitian ini digunakan metode Kontroler MPC untuk mengontrol gerak Translasi sumbu X, Y, dan Z.

### 4.3.1 Simulasi Pengujian Kontroler MPC pada Sumbu X

Simulasi pengujian kontroler MPC pada sumbu X dengan variasi nilai Np yang akan menghasilkan sinyal keluaran yang diprediksi akan berbeda karena banyaknya kondisi *state*. Variasi nilai Np ditunjukkan pada Tabel 4.1, dengan nilai Nc yang tetap. Pada Gambar 4.8 menunjukkan respon hasil dari simulasi dengan variasi Np dengan reference sinyal step.

**Tabel 4.1** Variasi Nilai Np pada Kontroler MPC sumbu X

| No. | Np | Nc | Kmpc | Ky |
|---|---|---|---|---|
| 1. | 18 | 2 | [1,459 ,0,115, 0,105] | 0,105 |
| 2. | 20 | 2 | [2,197, 0,191, 0,143] | 0,143 |
| 3. | 22 | 2 | [3,184, 0,302, 0,188] | 0,188 |




**Gambar 4.8** Respon Simulasi Sumbu X dengan Variasi Nilai Np

Dari hasil didapatkan bahwa semakin besar nilai dari Np yang digunakan maka respon dari sistem akan naik terus menuju steady state. Sedangkan semakin kecil nilai Np, respon semakin lambat menuju kondisi steady state. Variasi nilai Np ini juga berpengaruh pada nilai time konstan, semakin kecil nilai Np maka time konstan semakin besar.

Nilai Np yang dipilih adalah 18, dikarenakan respon tidak mengalami osilasi dan menuju steady state cukup cepat. Nilai time kontan, rise time, settling time dan over shoot dapat dilihat pada Tabel 4.2.

**Tabel 4.2** Parameter Hasil Tuning Variasi Nilai Np Kontroler MPC sumbu X

| No. | Np | Time Settling (±2%) (detik) | Rise Time (10%-90%) (detik) | Overshoot Maximum |
|---|---|---|---|---|
| 1. | 18 | 4,8 | 3 | 0,0008 |
| 2. | 20 | 4,6 | 1,5 | 0,00041 |
| 3. | 22 | 3,1 | 1 | 0,018 |

### 4.3.2 Simulasi Pengujian Kontroler MPC pada Sumbu Y

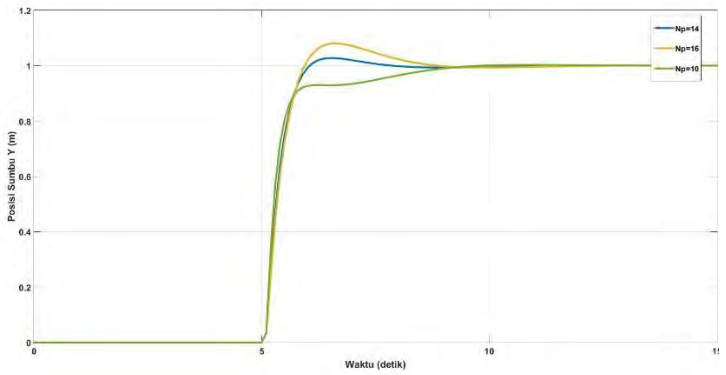
Pengujian dengan variasi nilai Np juga dilakukan untuk simulasi pengujian kontroler MPC pada sumbu Y dengan reference berupa sinyal

step dengan step time 5 detik. Variasi nilai Np ditunjukkan pada Tabel 4.3, dengan nilai Nc yang tetap.

**Tabel 4.3** Variasi Nilai Np pada Kontroler MPC sumbu Y

| No. | Np | Nc | Kmpc | Ky |
|---|---|---|---|---|
| 1. | 10 | 2 | [0,151 , 0,007, 0,019] | 0,019 |
| 2. | 14 | 2 | [0,55, 0,034, 0,05] | 0,05 |
| 3. | 16 | 2 | [0,923, 0,065, 0,074] | 0,074 |

Dari hasil didapatkan pada Gambar 4.9 bahwa pada saat Np bernilai 16, respon mengalami overshoot sekitar 0,1. Sedangkan pada Np bernilai 10 , respon tidak lambat untuk mencapai steady state. Sehingga dipilihlah nilai Np sebesar 14, walaupun masih terdapat overshoot sebesar kurang dari 0,05.Nilai rise time, settling time dan over shoot dapat dilihat pada tabel 4.4.




**Gambar 4.9** Respon Simulasi Sumbu Y dengan Variasi Nilai Np

**Tabel 4.4** Parameter Hasil Tuning Variasi Nilai Np Kontroler MPC sumbu Y

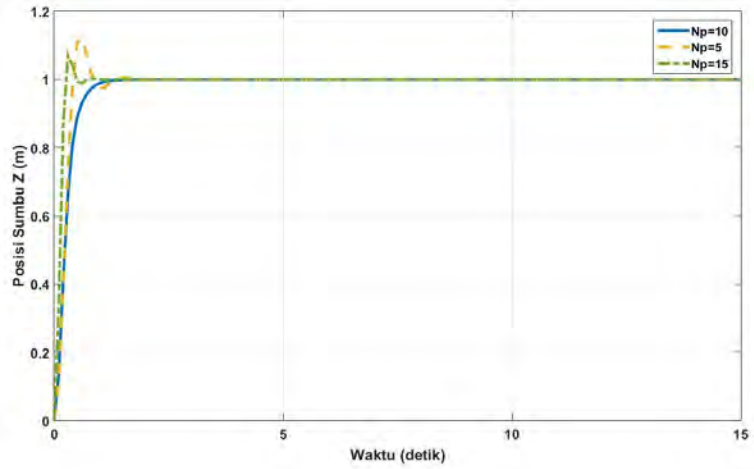
| No. | Np | Time Settling (±2%) (detik) | Rise Time (10%-90%) (detik) | Overshoot |
|---|---|---|---|---|
| 1. | 10 | 1,4 | 0,65 | 0,003 |
| 2. | 14 | 1,8 | 0,68 | 0,08 |
| 3. | 16 | 1,3 | 0,62 | 0,02 |

### 4.3.3 Simulasi Pengujian Kontroler MPC pada Sumbu Z

Simulasi pengujian kontroler MPC pada sumbu Z sama dengan pengujian kontroler MPC pada sumbu X maupun Y dengan menggunakan variasi nilai Np, dapat dilihat pada Tabel 4.5. Dari Gambar 4.10 didapatkan bahwa semakin kecil nilai Np maupun semakin besar, respon mengalami overshoot seperi pada saat Np sebesar 5 langkah dan Np sebesar 15 langkah. Pada Np sebesar 10, respon menuju pada reference tanpa mempunyai overshoot.

**Tabel 4.5** Variasi Nilai Np pada Kontroler MPC sumbu Z

| No. | Np | Nc | Kmpc | Ky |
|---|---|---|---|---|
| 1. | 5 | 5 | [0,833 , 0,22 , 0,196] | 0,196 |
| 2. | 10 | 5 | [6,90 , 3,23 , 0,878] | 0,878 |
| 3. | 15 | 5 | [7,734 , 4,882 , 0,734] | 0,734 |




**Gambar 4.10** Respon Simulasi Sumbu Z dengan Variasi Nilai Np

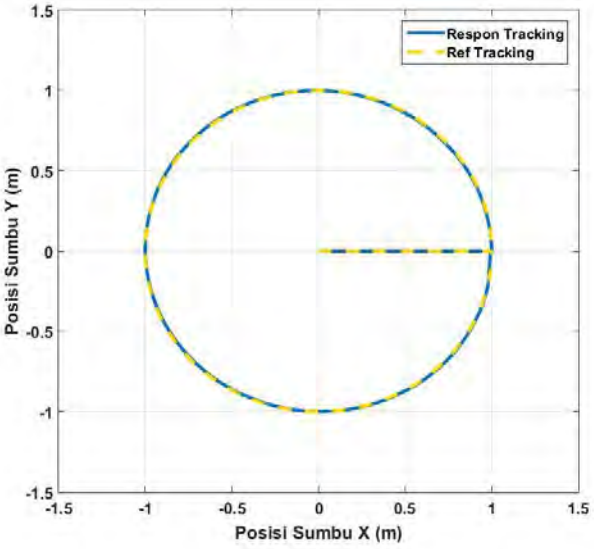
Variasi nilai Np ini juga berpengaruh pada nilai time konstan,rise time dan overshoot. Nilai Np untuk sumbu Z yang dipilih adalah 10, dikarenakan respon tidak mengalami overshoot dan waktu menuju steady state yang cukup cepat. Nilai-nilai hasil tuning variasi Np bisa dilihat pada Tabel 4.6

**Tabel 4.6** Parameter Hasil Tuning Variasi Nilai Np Kontroler MPC sumbu Z

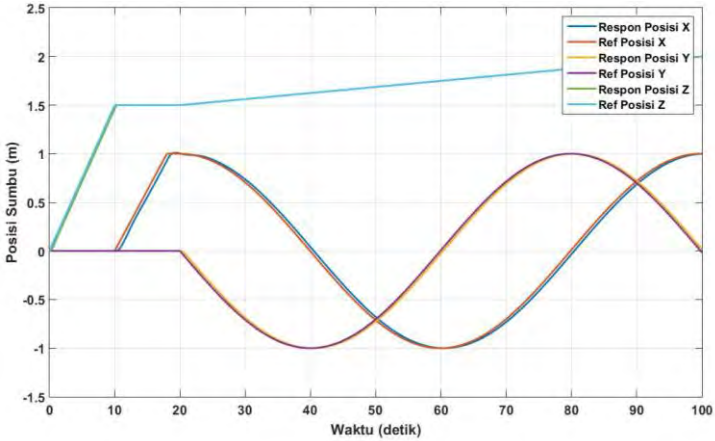
| No. | Np | Time Settling (±2%) (detik) | Rise Time (10%-90%) (detik) | Overshoot |
|---|---|---|---|---|
| 1. | 5 | 1,1 | 0,27 | 0,12 |
| 2. | 10 | 1,12 | 0,5 | - |
| 3. | 15 | 0,6 | 0,18 | 0,07 |

## 4.4 Simulasi Pengujian *Trajectory Tracking Quadcopter* pada Lintasan Lingkaran dan Segiempat.

Referensi jalur yang akan disimulasikan untuk *trajectory tracking quadcopter* berbentuk lingkaran, dengan jari-jari 1 m, kecepatan sudut 0,1 rad/s, respon terlihat pada Gambar 4.11. Sedangkan untuk grafik respon posisi quadcopter terhadap waktu dapat dilihat pada Gambar 4.12.




**Gambar 4.11** Respon Simulasi *Quadcopter Trajectory tracking* dengan Lintasan Lingkaran

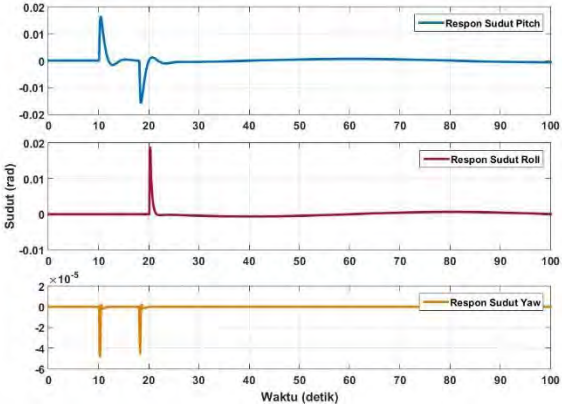



**Gambar 4.12** Respon Pengujian *Trajectory tracking Quadcopter* pada Lintasan Lingkaran pada Posisi Sumbu X, Y dan Z

Waktu simulasi yang digunakan selama 200 detik. Pada detik ke 0 sampai ke 10 *quadcopter* melakukan take off pada koordinat (0,0,0) hingga pada koordinat (0,0,1.5). Selanjutnya pada detik ke 10 sampai detik ke 10, *quadcopter* bergerak ke samping pada koordinat (1,0,2). *Quadcopter* dalam keaadan hover selama 2 detik pada detik ke 18 sampai detik ke 20. Setelah itu, pada detik ke 20, *quadcopter* mulai menjejak lintasan lingkaran sampai detik ke 100, dengan referensi sumbu z terus naik sampai ketinggian 2 meter.

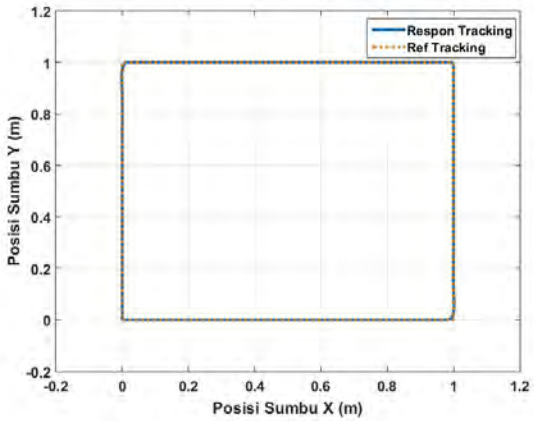
Pada Gambar 4.11 dapat dilihat bahwa *quadcopter* mampu mengikuti lintasan lingkaran yang diberikan namun *quadcopter* masih mempunyai undershoot sekitar 0,025. Pada koordinat (0.955,-0.155,2), *quadcopter* mampu kembali ke lintasan.

Respon pada gerak rotasi dapat dilihat pada Gambar 4.13. Respon yang diharapkan untuk gerak rotasi adalah nilai sudut *pitch* , *roll* dan *yaw* tetap stabil bernilai 0. Dapat dilihat pada respon sudut *pitch*, sudut *roll* maupun sudut *yaw* masih terdapat osilasi namun sangat kecil sehingga masih dianggap mendekati 0 dan tidak berpengaruh pada gerak Translasi.



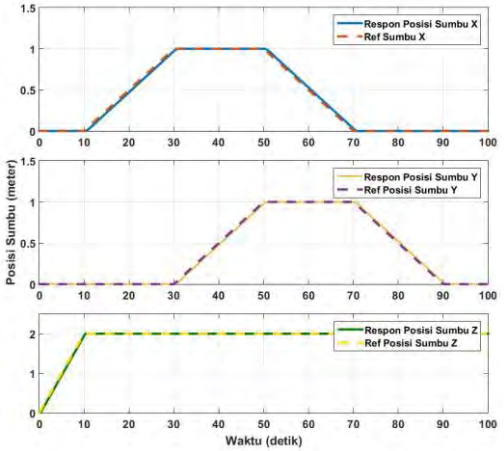

**Gambar 4.13** Respon Sudut Pengujian *Trajectory tracking Quadcopter* pada Lintasan Lingkaran

Selanjutnya simulasi pengujuan *trajectory tracking* menggunakan lintasan segiempat, terlihat pada Gambar 4.14. Waktu simulasinya sebesar 100 detik. pada koordinat (0,0,0), *quadcopter* melakukan take-off vertical selama 10 detik sampai ketinggian 1 meter, selanjutnya *quadcopter* menjejak lintasan segiempat selama 80 detik. setelah itu *quadcopter* melakukan hover pada koordinat awal selama 10 detik.




**Gambar 4.14** Respon Simulasi *Quadcopter Trajectory tracking* dengan Lintasan Segiempat

Dapat dilihat pada Gambar 4.14, *quadcopter* masih belum menjejak referensi pada setiap sudut segiempat dikarekan perubahan posisi yang sangat cepat. Grafik respon posisi quadcopter saat melakukan tracking segiempat dapat dilihat pada Gambar 4.15.

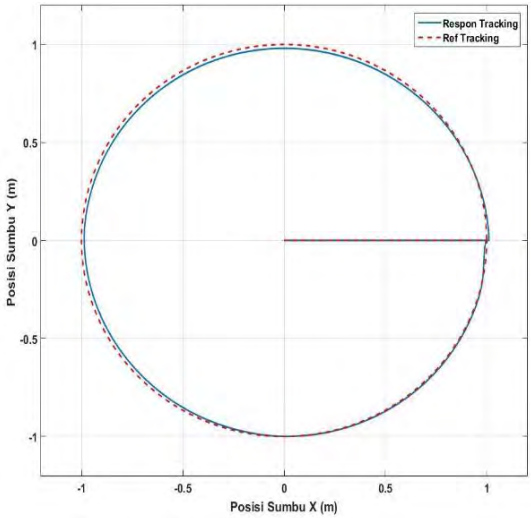



**Gambar 4.15** Respon Pengujian *Trajectory tracking Quadcopter* pada Lintasan Segiempat pada Posisi Sumbu X, Y dan Z

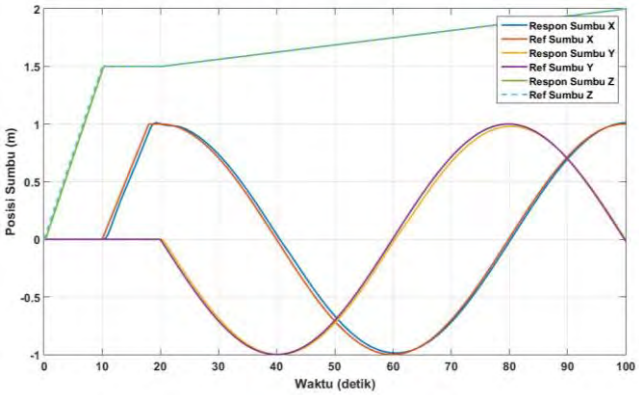
## 4.5 Simulasi Pengujian *Trajectory tracking Quadcopter* pada Lintasan Lingkaran dan Segiempat Dengan *Disturbance*

Pada simulasi pengujian *trajectory tracking quadcopter* pada lintasan lingkaran dan segiempat dengan *disturbance* menggunakan sinyal step yang ditempatkan pada sinyal kontrol U2, U3 dan U4. *Disturbance* ini diasumsikan sebagai gangguan seperti angin yang dapat membuat *quadcopter* terdorong keluar lintasan. Gangguan pada sinyal kontrol U2,U3 dan U4 mempengaruhi posisi sudut dari *quadcopter*. Gangguan pada sinyal kontrol U2 dan U3 juga dapat mempengaruhi posisi *quadcopter* sumbu X dan Y.Diberikan gangguan sinyal kontrol U2 sebesar 5 pada nilai final value sinyal step pada detik ke 60, gangguan sinyal kontrol U3 sebesar 0.1 pada detik ke 40 dan gangguan sinyal kontrol U4 sebesar 1 pada detik ke 30.

Pengujian *trajectory tracking* dengan *disturbance* pada lintasan lingkaran ditunjukkan pada Gambar 4.16 dan Gambar 4.17.




**Gambar 4.16** Respon Simulasi *Quadcopter Trajectory tracking* pada Lintasan Lingkaran dengan *disturbance*

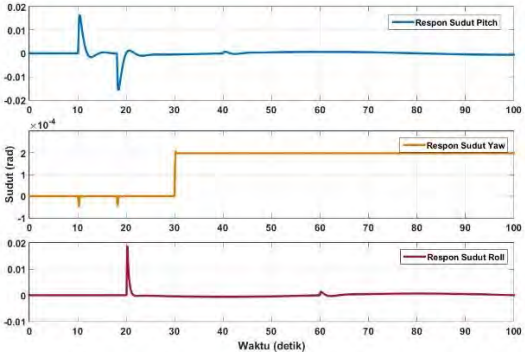



**Gambar 4.17** Respon Posisi Sumbu X dan Y *Quadcopter* pada Lintasan Lingkaran dengan *disturbance*

Pada pengujian trajectory tracking quadcopter untuk lintasan lingkaran, didapatkan bahwa saat diberi gangguan, terdapat error yang cukup besar pada detik ke 40, error tersebut dikarenakan gangguan pada

sinyal kontrol U2 dan dapat membuat *quadcopter* keluar lintasan namun hanya sesaat. Pada gerak sumbu X memiliki RMSE sebesar 3,34% dan pada gerak sumbu Y memiliki RMSE sebesar 2,38%.

Dapat dilihat pada Gambar 4.18, gangguan pada sinyal kontrol mempengaruhi respon sudut *quadcopter*. Gangguan pada sinyal kontrol U2 mempengaruhi respon sudut *roll*, gangguan pada sinyal kontrol U3 mempengaruhi respon sudut *pitch* sedangkan gangguan pada sinyal kontrol U4 mempengaruhi sudut *yaw*.

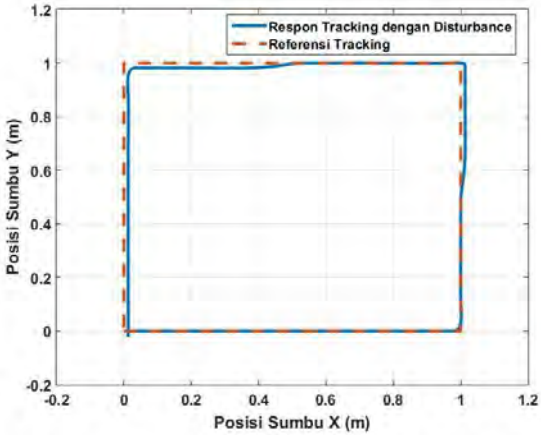



**Gambar 4.18** Respon Sudut *Quadcopter* pada Lintasan Lingkaran dengan *disturbance*

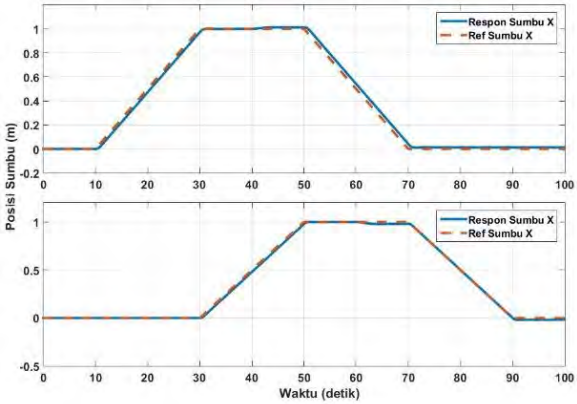
Pengujian kedua adalah dengan lintasan segiempat dengan *disturbance* yang sama dengan lintasan lingkaran, didapatkan bahwa saat diberi gangguan, terdapat error yang cukup besar pada detik ke 40, error tersebut dikarenakan gangguan pada sinyal kontrol U3 dan dapat membuat *quadcopter* keluar lintasan namun hanya sesaat. Pada gambar tersebut, sistem ketika dalam gangguan mengalami *error* yang cukup besar, namun masih dapat mengikuti referensi yang diberikan. Namun pada detik ke 60, gangguan yang disebabkan sinyal kontrol U2 juga terdapat error yang menggeser *quadcopter* dari jalur referensinya.

Pada trakcing lintasan segiempat, gerak sumbu X *quadcopter* memiliki RMSE sebesar 2,73% dan pada gerak sumbu Y *quadcopter* memiliki RMSE sebesar 1,25%. Respon sumbu X dan Y pada lintasan segiempat terhadap gangguan dapat dilihat pada Gambar 4.19 dan respon sudut *quadcopter* untuk tracking segiempat dapat dilihat pada Gambar

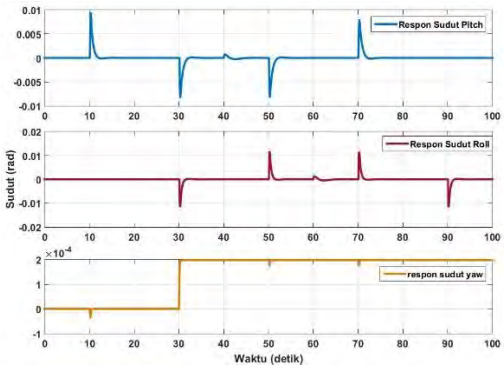
4.20. Respon sudut quadcopter untuk pengujian disturbance dengan lintasan segiempat dapat dilihat pada Gambar 4.21




**Gambar 4.19** Respon Simulasi *Quadcopter Trajectory tracking* pada Lintasan Lingkaran dengan *disturbance*



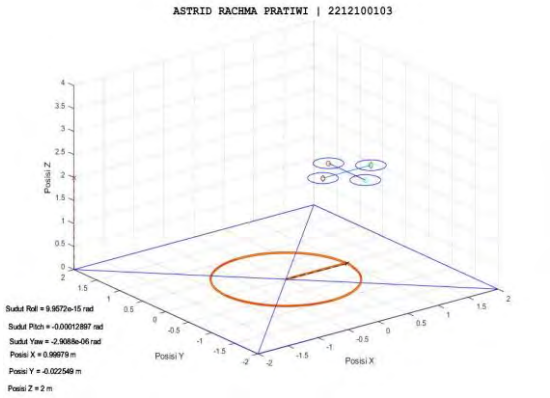

**Gambar 4.20** Respon Posisi Sumbu X dan Y *Quadcopter* pada Lintasan Lingkaran dengan *disturbance*



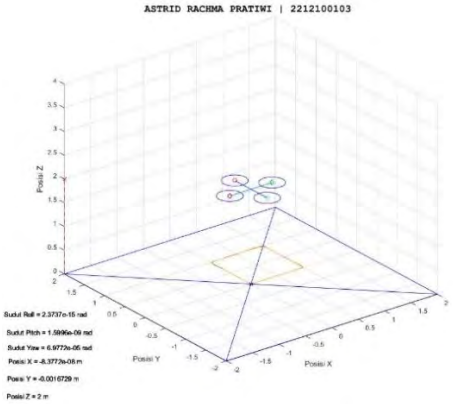

**Gambar 4.21** Respon Sudut *Quadcopter* pada Lintasan Segiempat dengan *disturbance*

## 4.6 Simulasi 3D

Pada simulasi ini digunakan simulasi 3D untuk mengetahui pergerakan *quadcopter* yang sesungguhnya. Simulasi 3D ini dirancang menggunakan lintasan berbentuk lingkaran. Pada sumulasi 3D dibatasi sumbu *X* dan *Y* bernilai 2 meter dan sumbu Z bernilai 4 meter untuk mempermudah dalam pembuatan simulasi 3D. simulasi 3D ditunjukkan pada Gambar 4.22 dan simulasi 3D dengan lintasan segiempat ditunjukkan pada Gambar 4.23.




**Gambar 4.22** Simulasi 3D *Trajectory tracking* Lintasan Lingkaran pada *Quadcopter*




**Gambar 4.23** Simulasi 3D *Trajectory tracking* Lintasan Segiempat pada *Quadcopter*

*[Halaman ini sengaja dikosongkan]*

# LAMPIRAN A

## A1. Data Pengukuran Kecepatan Motor

| Pulsa | Motor 1 | Motor 2 | Motor 3 | Motor 4 |
|---|---|---|---|---|
| 75 | 114.505 | 111.888 | 109.376 | 113.668 |
| 80 | 152.29 | 152.604 | 148.836 | 154.069 |
| 85 | 187.563 | 192.900 | 187.772 | 193.633 |
| 90 | 219.276 | 225.242 | 221.684 | 229.429 |
| 95 | 252.77 | 258.108 | 256.747 | 263.76 |
| 100 | 281.03 | 290.973 | 287.31 | 295.578 |
| 105 | 313.267 | 325.722 | 322.373 | 330.223 |
| 110 | 342.26 | 355.552 | 351.156 | 360.262 |
| 115 | 369.892 | 384.964 | 380.986 | 390.93 |
| 120 | 399.198 | 416.468 | 412.386 | 422.958 |
| 125 | 427.04 | 447.031 | 440.646 | 452.16 |
| 130 | 453.625 | 471.732 | 466.29 | 475.396 |
| 135 | 479.164 | 500.411 | 496.957 | 508.575 |
| 140 | 502.295 | 503.97 | 519.042 | 533.172 |
| 145 | 525.531 | 523.752 | 543.848 | 558.292 |
| 150 | 546.569 | 548.872 | 566.037 | 582.784 |
| 155 | 570.538 | 572.212 | 574.410 | 606.334 |
| 160 | 590.529 | 595.344 | 592.832 | 630.721 |
| 165 | 611.567 | 615.126 | 609.997 | 654.271 |
| 170 | 633.024 | 629.465 | 627.790 | 656.050 |
| 175 | 651.236 | 653.434 | 650.817 | 661.702 |
| 180 | 673.111 | 670.076 | 672.378 | 673.844 |

| Pulsa | Motor 1 | Motor 2 | Motor 3 | Motor 4 |
|---|---|---|---|---|
| 185 | 676.356 | 672.169 | 675.937 | 676.565 |
| 190 | 673.006 | 672.483 | 673.948 | 675.832 |
| 195 | 672.902 | 674.053 | 673.00 | 675.1 |
| 200 | 673.006 | 674.262 | 672.797 | 675.623 |

## A2 Data Pengukuran Gaya Angkat Motor

| Pulsa | Gaya Angkat 1 (kg) | Gaya Angkat 2 (kg) | Gaya Angkat 3 (kg) | Gaya Angkat 4 (kg) |
|---|---|---|---|---|
| 75 | 0.02 | 0.011 | 0.017 | 0.014 |
| 80 | 0.036 | 0.031 | 0.033 | 0.039 |
| 85 | 0.058 | 0.055 | 0.055 | 0.061 |
| 90 | 0.078 | 0.08 | 0.077 | 0.084 |
| 95 | 0.104 | 0.107 | 0.102 | 0.115 |
| 100 | 0.131 | 0.138 | 0.134 | 0.145 |
| 105 | 0.165 | 0.177 | 0.17 | 0.184 |
| 110 | 0.198 | 0.207 | 0.209 | 0.221 |
| 115 | 0.227 | 0.252 | 0.243 | 0.264 |
| 120 | 0.27 | 0.297 | 0.285 | 0.306 |
| 125 | 0.31 | 0.34 | 0.328 | 0.357 |
| 130 | 0.352 | 0.39 | 0.374 | 0.402 |
| 135 | 0.396 | 0.432 | 0.426 | 0.452 |
| 140 | 0.438 | 0.446 | 0.478 | 0.498 |
| 145 | 0.488 | 0.475 | 0.522 | 0.553 |
| 150 | 0.533 | 0.54 | 0.578 | 0.611 |
| 155 | 0.582 | 0.605 | 0.613 | 0.664 |

| Pulsa | Gaya Angkat 1 (kg) | Gaya Angkat 2 (kg) | Gaya Angkat 3 (kg) | Gaya Angkat 4 (kg) |
|---|---|---|---|---|
| 160 | 0.63 | 0.626 | 0.619 | 0.718 |
| 165 | 0.672 | 0.654 | 0.646 | 0.771 |
| 170 | 0.723 | 0.715 | 0.704 | 0.783 |
| 175 | 0.772 | 0.784 | 0.762 | 0.791 |
| 180 | 0.834 | 0.832 | 0.826 | 0.825 |
| 185 | 0.842 | 0.846 | 0.84 | 0.847 |
| 190 | 0.833 | 0.833 | 0.836 | 0.841 |
| 195 | 0.832 | 0.834 | 0.834 | 0.845 |
| 200 | 0.834 | 0.836 | 0.837 | 0.845 |

**A3. Pengukuran Kecepatan Motor**

Pengukuran ini dilakukan dengan menggunakan tachometer laser dan menggunakan sinyal masukan berupa PWM dari Arduino. hasil pengukuran kecepatan motor ditampilkan pada Lampiran A1. Dari hasil pengukuran diperoleh persamaan linear hubungan pulsa motor terhadap kecepatan (rad/s) pada masing-masing motor terdapat pada Persamaan (1).

$$y_1 = 44,851x - 1760,5$$
$$y_2 = 44,10x - 1603,7$$
$$y_3 = 44,8x - 1670,4$$
$$y_4 = 45,047x - 1618,7$$

**A4. Pengukuran Gaya Angkat Motor**

Langkah-langkah dalam pengukuran gaya angkat motor adalah dengan meletakkan motor dan propeller tepat diatas beban sehingga memiliki berat total 1,618 kg. Beban tersebut merupakan botol aqua 1,5 liter dengan diisi air, terdapat pada Gambar 3.4. Kemudian diletakkan diatas timbangan digital dan motor dijalankan dengan memberikan pulsa atau sinyal PWM dari arduino. Hasil pembacaan berat pada timbangan

dikurangi dengan beban 1,618 kg adalah gaya angkat pada masing-masing motor. Pada eksperimen ini dilakukan pengukuran gaya angkat pada masing-masing motor *propeller*.Dari pengukuran diperoleh persamaan linear hubungan pulsa motor terhadap gaya angkat kg pada masing-masing motor terdapat pada Persamaan berikut :

$$y_1 = 0,0076x - 0,6097$$
$$y_2 = 0,0076x - 0,5968$$
$$y_3 = 0,0076x - 0,5968$$
$$y_4 = 0,0078x - 0,6026$$

**A4. Gaya thrust**

Untuk mendapatkan konstanta *thrust, quadcopter* diterbangkan dalam posisi *hover* karena pada posisi tersebut, gaya yang dihasilkan oleh motor untuk mengangkat *quadcopter* sama dengan gaya yang mendorong *quadcopter* menuju *ground*. Dengan asumsi gaya gravitasi tetap, maka konstanta *thrust* dapat dihitung menggunakan Persamaan dibawah ini :

$$b = \frac{m.g}{\sum_{i=1}^{4} \Omega_i^{\,2}}$$

Konstanta *drag* dihitung dengan persamaan gerak lurus berubah beraturan. Pengukuran konstanta *drag* dilakukan dengan mengambil data penerbangan *quadcopter* pada saat *take-off*. Gaya yang terjadi saat *quadcopter* bergerak ke atas dapat diperoleh menggunakan Persamaan dibawah ini :

$$\sum F = Thrust - mg - Drag$$

$$ma = (\tau_1 + \tau_2 + \tau_3 + \tau_4) - mg - D$$

$$D = ma - ((\tau_1 + \tau_2 + \tau_3 + \tau_4) - mg)$$

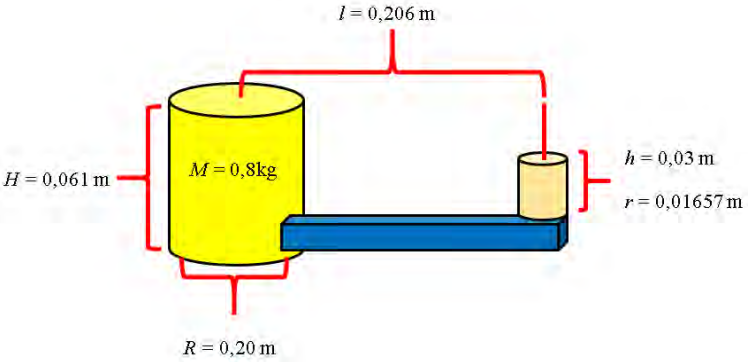
$$a = \frac{s - v_0 t}{0,5t^2}$$

$$CD = \frac{D}{\Omega^2}$$

Dimana:

$CD$ = Konstanta *Drag*
$v_0$ = Kecepatan awal ( $m/s$ )
$t$ = Waktu ( $s$ )
$a$ = Percepatan ( $m/s^2$ )
$s$ = Jarak yang ditempuh ( $m$ )
$\tau_i$ = Gaya angkat, i=1,2,3,4
$D$ = Gaya *drag*
$\Omega$ = Kecepatan motor

**A5. Identifikasi Fisik *Quadcopter***




Identifikasi fisik konstanta

$$I_{z,frame1,4} = \frac{m_l}{12}(p^2 + l^2)$$

$$I_{z.frame2,3} = \frac{m_l}{12}(p^2 + t^2)$$

$$I_{z.m1,2,3,4} = \frac{m_m}{2} r^2 + (p_1)^2 . M$$

$$I_{z.batt} = \frac{m_b}{12}(p^2 + l^2)$$

$$I_{z.1} = \frac{m_{mika}}{12}(p^2 + l^2)$$

$$I_{z.2} = \frac{m_{mika}}{12}(2.l^2)$$

$$I_{z.1,2,3,4} = \frac{m_p}{2} r^2 + (p_1)^2 . M_p$$

$$I_{x.frame1,2,3} = \frac{m_l}{12}(l^2 + t^2)$$

$$I_{x.frame4,5,6} = \frac{m_l}{12}(p^2 + l^2)$$

$$I_{z.m2,4} = m_m \left( \frac{r^2}{4} + \frac{h}{12} \right) + [(0.5h)^2 + (p_1 - p_2)^2]\, m$$

$$I_{z.m1,3} = m_m \left( \frac{r^2}{4} + \frac{h}{12} \right) + (0.5h)^2\ m$$

$$I_{x.batt} = \frac{m_b}{12}(l^2 + t^2) + (0.5t)^2$$

$$I_{y.batt} = \frac{m_b}{12}(p^2 + t^2) + (0.5t)^2$$

$$I_{x.mika1} = \frac{m_{mika}}{12}(p^2 + t^2) + (t^2)\ m_{mika}$$

$$I_{x.mika2,3} = \frac{m_{mika}}{12}(l^2 + t^2) + (t^2)\ m_{mika}$$

$$I_{y.mika1} = \frac{m_{mika}}{12}(l^2 + t^2) + (t^2)\, m_{mika}$$

$$I_{x.mika2,3} = \frac{m_{mika}}{12}(l^2 + t^2) + (t^2 + l^2)\, m_{mika}$$

$$I_{x.1,2} = m_p \left( \frac{{p_2}^2}{6} + \frac{t^2}{12} + {p_1}^2 + (0.5t^2 + h)^2 \right)$$

$$I_{x.3,4} = m_p \left( \frac{{p_2}^2}{6} + \frac{t^2}{12} \right)$$

Perhitungan $I_{xx}$, $I_{yy}$, $I_{zz}$ dan $J_{TP}$

$$I_{xx} = \frac{mr^2}{4} + \frac{mh^2}{6} + 2mr^2 + \frac{MR^2}{4} + \frac{MH^2}{12}$$

$I_{xx} = 1{,}68\text{x}10^{-3}$

$$I_{yy} = \frac{mr^2}{4} + \frac{mh^2}{6} + 2mr^2 + \frac{MR^2}{4} + \frac{MH^2}{12}$$

$I_{yy} = 1{,}68\text{x}10^{-3}$

$$I_{zz} = \frac{MR^2}{2} + 4mr^2$$

$Izz = 1{,}25\text{x}10^{-3}$

$$J_{TP} = \frac{4 \times \left(mr^2\right)}{2}$$

$J_{TP} = 0{,}21175\text{x}10^{-4}$

# LAMPIRAN B

## B1. Program Plotter *Quadcopter*

```
function [sys,x0,str,ts] =
% PARAMETERS
% s defines the plot size in meters
% swi controls flyer attitude plot; 1 = on,
otherwise off.
% INPUTS
% 1 Center X position
% 2 Center Y position
% 3 Center Z position
% 4 Yaw angle in rad
% 5 Pitch angle in rad
% 6 Roll angle in rad
% OUTPUTS
%   None
ts = [-1 0];

switch flag,
  case 0
    [sys,x0,str,ts] =
mdlInitializeSizes(ts,plot,enable); %
Initialization
  case 3
    sys = mdlOutputs(t,u,s,plot,enable); %
Calculate outputs
  case {1,2, 4, 9} % Unused flags
    sys = [];
  otherwise
    error(['unhandled flag =
',num2str(flag)]); % Error handling
end

% Initialize
function [sys,x0,str,ts] =
mdlInitializeSizes(ts,plot,enable)
% Call simsizes for a sizes structure, fill
it in, and convert it
```

```
% to a sizes array.
sizes = simsizes;
sizes.NumContStates  = 0;
sizes.NumDiscStates  = 0;
sizes.NumOutputs     = 0;
sizes.NumInputs      = 6;
sizes.DirFeedthrough = 1;
sizes.NumSampleTimes = 1;
sys = simsizes(sizes);
x0  = [];
str = [];          % Set str to an empty
matrix.
ts = [0.5 0];
if enable == 1
    figure(plot);
    clf;
    %colordef(1,'none');
end
% End of mdlInitializeSizes.
function sys = mdlOutputs(t,u,s,plot,enable)
global a1s b1s
    name =
strcat('flyer_movie',num2str((t/0.125)*2+1),'
r.bmp'); %odds
    %name = strcat('flyer_movie',num2str(160-
(t/0.125)*2),'l.bmp'); %evens
if size(a1s) == [0 0];
    a1s = [0 0 0 0];
    b1s = [0 0 0 0];
end
d = 0.40; %Jarak tiap pusat motor ke motor
lainnya
r = 0.206; %Jarak motor ke pusat quadcopter
N = 1;
S = 2;
E = 3;
W = 4;
D(:,N) = [d;0;0]; %displacements
D(:,S) = [-d;0;0];
D(:,E) = [0;d;0];
```

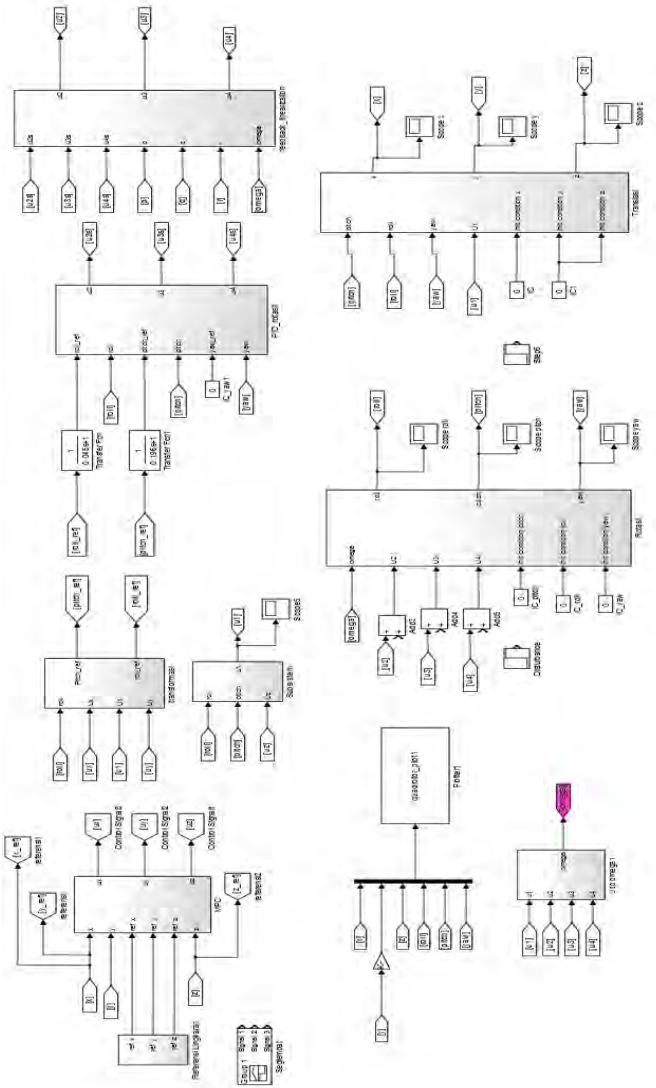
```
D(:,W) = [0;-d;0];
C(:,N) = [s(1)/4;0;0]; %Attitude center
displacements
C(:,S) = [-s(1)/4;0;0];
C(:,E) = [0;s(1)/4;0];
C(:,W) = [0;-s(1)/4;0];
if enable == 1
    %draw ground
    figure(plot);
    clf;
    if length(s) == 1
        axis([-s s -s s 0 s]);
    else
        axis([-s(1) s(1) -s(1) s(1) 0 s(2)])
        s = s(1);
    end
    hold on;
    plot3([-s -s],[s -s],[0 0],'-b')
    plot3([-s s],[s s],[0 0],'-b')
%garisbiruWEST
    plot3([s -s],[-s -s],[0 0],'-b')
    plot3([s s],[s -s],[0 0],'-
b')%garisbiruNORTH
    plot3([s -s],[-s s],[0 0],'-b')
%lintanglurus
    plot3([-s s],[-s s],[0 0],'-b')
%lintangsilang
    %READ STATE
    z = [u(1);u(2);-u(3)];
    n = [u(4);u(5);u(6)];
    %PREPROCESS ROTATION MATRIX
    phi = n(1);    %Euler angles
    the = n(2);
    psi = n(3);

    R = [cos(the)*cos(phi)
sin(psi)*sin(the)*cos(phi)-cos(psi)*sin(phi)
cos(psi)*sin(the)*cos(phi)+sin(psi)*sin(phi);
%BBF > Inertial rotation matrix
```

```
         cos(the)*sin(phi)
sin(psi)*sin(the)*sin(phi)+cos(psi)*cos(phi)
cos(psi)*sin(the)*sin(phi)-sin(psi)*cos(phi);
         -sin(the)          sin(psi)*cos(the)
cos(psi)*cos(the)];
        %Manual Construction
        %Q3 = [cos(psi) -sin(psi) 0;sin(psi)
cos(psi) 0;0 0 1];   %Rotation mappings
        %Q2 = [cos(the) 0 sin(the);0 1 0;-
sin(the) 0 cos(the)];
        %Q1 = [1 0 0;0 cos(phi) -sin(phi);0
sin(phi) cos(phi)];
        %R = Q3*Q2*Q1;    %Rotation matrix

    %CALCULATE FLYER TIP POSITONS USING
COORDINATE FRAME ROTATION
  F = [1 0 0;0 -1 0;0 0 -1];
    %Draw flyer rotors
    t = [0:pi/8:2*pi];
for j = 1:length(t)
        circle(:,j) =
[r*sin(t(j));r*cos(t(j));0];
    end
    title('ASTRID RACHMA PRATIWI | 2212100103
','FontSize',15,'FontName','Courier
New','FontWeight','bold');
    for i = [N S E W]
        hub(:,i) = F*(z + R*D(:,i)); %points
in the inertial frame
        q = 1; %Flapping angle scaling for
output display - makes it easier to see what
flapping is occurring
        Rr = [cos(q*a1s(i))
sin(q*b1s(i))*sin(q*a1s(i))
cos(q*b1s(i))*sin(q*a1s(i));   %Rotor > Plot
frame
              0              cos(q*b1s(i))
-sin(q*b1s(i));
```

```
                -sin(q*a1s(i))
sin(q*b1s(i))*cos(q*a1s(i))
cos(q*b1s(i))*cos(q*a1s(i))];
        tippath(:,:,i) = F*R*Rr*circle;
plot3([hub(1,i)+tippath(1,:,i)],[hub(2,i)+tip
path(2,:,i)],[hub(3,i)+tippath(3,:,i)],'b-')
        tinggi=u(3);
        roll_plot= u(6);
        pitch_plot= u(5);
        maju=u(1);
        minggir=u(2);
        yaw_plot=u(4);
        if tinggi<0.05
            tinggi=0;
        end
        if roll_plot<0.0001
            roll_plot= u(6);
        end
        if pitch_plot<0.0001
            pitch_plot= u(5);
        end
        textroll=text(-3.6,1.4,['Sudut Roll =
', num2str(roll_plot),' rad']);
        textpitch=text(-3.975,0.85,['Sudut
Pitch = ', num2str(pitch_plot),' rad']);
        textyaw=text(-4.3,0.4,['Sudut Yaw =
', num2str(yaw_plot),' rad']);
        textmaju=text(-4.6,0,['Posisi X = ',
num2str(maju),' m']);
        textminggir=text(-5,-0.5,['Posisi Y =
', num2str(-minggir),' m']);
        textheight=text(-5.4,-1,['Posisi Z =
', num2str(tinggi),' m']);



    end
    %Draw flyer
    plot3([hub(1,N) hub(1,S)],[hub(2,N)
hub(2,S)],[hub(3,N) hub(3,S)],'-d')
```

```
    plot3([hub(1,E) hub(1,W)],[hub(2,E)
hub(2,W)],[hub(3,E) hub(3,W)],'-b')

plot3([hub(1,N)],[hub(2,N)],[hub(3,N)],'og')

plot3([hub(1,S)],[hub(2,S)],[hub(3,S)],'or')

plot3([hub(1,E)],[hub(2,E)],[hub(3,E)],'oc')
%isi lingkaran dalam

plot3([hub(1,W)],[hub(2,W)],[hub(3,W)],'or')

    %Tracking lines
    plot3([z(1) 0],[-z(2) 0],[0 0],'--k')
%titik2trajektoriX
    plot3([z(1)],[-z(2)],[0],'xk')
%titik2trajektoriY
    plot3([-s -s],[s s],[-z(3) 0],'--r')
%titik2trajektoriZ
    plot3([-s],[s],[-z(3)],'xr')
    xlabel('Posisi X');
    ylabel('Posisi Y');
    zlabel('Posisi Z');
    grid on
%mov=avifile('quad.avi','Compression','None',
'Quality',100,'fps',10)
    %video=figure(plot);
end
%saveas(1,name);
sys = [];
% End of mdlOutputs.
```

**B2. Diagram *Simulink***



# BAB V
# PENUTUP

Setelah rangkaian penentuan parameter dan perancangan yang telah dilakukan dianalisis maka akan dapat ditarik kesimpulan. Pembahasan dari bab-bab sebelumnya dan kendala-kendala yang terjadi selama pengerjaan tugas akhir ini akan menjadi bahan pertimbangan atau referensi dalam melakukan penelitian pengembangan dari tugas akhir ini.

## 5.1 Kesimpulan

Berdasarkan dari pengerjaan dan analisis data pada tugas akhir ini dapat diambil beberapa kesimpulan sebagai berikut :

a. Kontroler MPC yang digunakan bergantung pada pemodelan dan linearisasi *Plant*.
b. Pengendalian *quadcopter* membutuhkan respon rotasi yang lebih cepat dari pada respon translasi.
c. Respon *Quadcopter* yang dikendalikan dengan kontroler MPC saat melakukan *trajectory tracking* khususnya gerakan *cruise* dapat mengikuti referensi tracking yang diberikan dengan parameter kontroler Np_x sebesar 20, Np_y sebesar 14 dan Np_z sebesar 10, walaupun masih terdapat lagging pada setiap sumbu.

## 5.2 Saran

Untuk membentuk suatu sistem monitoring yang lebih handal, efektif, dan efisien, maka sebaiknya perlu ada improvisasi dan pengembangan lebih dari tugas akhir ini. Maka dari itu berikut beberapa saran yang ditujukan untuk tugas akhir ini :

a. Pemahaman teknis berupa hal-hal mekanik, elektrik dan ilmu pemrograman dalam implementasi quadrotor sangat dibutuhkan.
b. Diperlukan peralatan atau prosedur untuk memvalidasi momen inersia dari quadrotor, karena hal tersebut akan sangat berpengaruh pada validasi model matematika.
c. Kendala utama penggunaan kontroler PID pada *plant* kompleks adalah mencari metode *tunning* yang tepat, untuk itu diperlukan metode *tunning* yang lebih baik.

*[Halaman ini sengaja dikosongkan]*

# DAFTAR PUSTAKA


[1] H. Akhmad, P. U. Gilang, dan O. Hary, "Sistem Kontrol Altitude Pada UAV Model *Quadcopter* Dengan Metode PID", The 14th Industrial Electronics Seminar, Electronic Engineering Polytechnic Institute of Surabaya (EEPIS), Indonesia, October, 2012

[2] Chipofya. Mapopa, Jin Lee. Deok, dan Kil To Chong, "*Trajectory tracking* and Stabilization of a Quadrotor Using *Model Predictive Control* of Laguerre Function," Research Article Republic of Korea. 2014.

[3] J. M. Maciejowski, Predictive *Control* with Constraints, Prentice Hall, NewYork, NY,USA, 1st edition, 2002.

[4] L. Wang, "Discrete time *Model Predictive Control* design using Laguerre functions," in Proceedings of the American *Control* Conference, pp. 2430–2435, Arlington, Va, USA, June 2001.

[5] Tommaso Bresciani, "Modelling, Identification and *Control* of a Quadrotor Helicopter". Department of Automatic *Control* Lund University, October 2008.

[6] Teppo Luukkonen, "Modelling and *control* of *Quadcopter*". School of Science Aalto University, August 2011.

[7] H.K. Khalil and JW Grizzle, "*Nonlinear systems",* Prentice Hall, New Jersey, 2002

[8] Syamsudduha Syahrorini, "Analisis dan Simulasi PID Adaptif Optimal untuk Pengaturan Kecepatan dengan menggunakan Flexible Transmision", Tesis, Institut Teknologi Sepuluh Nopember, Surabaya, 2008

[9] Astrom, K., Hagglund, T. , "*PID Controllers: Theory, Design, and Tunning*", Instrument Society of America, 1995.

[10] Yuliansyah, Rendy.,*"Desain dan Implementasi Kontroler PID untuk Glide-Scope Tracking pada Fixed-Wing UAV (Unmanned Aerial Vehicle)*. Surabaya: Jurusan Teknik Elektro, Institut Teknologi Sepuluh Nopember, 2011.

[11] Wang, L., "*Model Predicitive Control System Design and Implementation Using MATLAB*", Springer, London, 2009.

[12] Akbar, Farid Choirul, "*Tuning Parameter Linier Quadritic Tracking Menggunakan Algoritma Genetika Untuk Pengendalian Gerak Lateral Quadcopter*", Surabaya: Jurusan Teknik Elektro, Institut Teknologi Sepuluh Nopember.2016

*[Halaman ini sengaja dikosongkan]*

# BIOGRAFI PENULIS

Astrid Rachma Pratiwi merupakan salah satu mahasiswa ITS yang mengambil jurusan teknik elektro. Fokusan yang diambil dalam menjalani kuliah adalah Sistem Pengaturan. Ia yang berkelahiran di Surabaya, 14 April 1994 merupakan anak ke 3 dari 3 bersaudara dari pasangan Ketut Ponco Margono dan Dwi Astuti. Lulus dari SD Muhammdiyah 15 Surabaya pada tahun 2006, kemudian melanjutkan studi ke jenjang lebih lanjut di SMPN 16 Surabaya dan lulus pada tahun 2009. Kemudian melanjutkan ke SMAN 15 Surabaya dan lulus pada tahun 2012. Setelah menembuh studi pada tingkat SMA, penulis melanjutkan ketingkat lebih lanjut, yaitu di Institut Teknologi Sepuluh Nopember Surabaya jurusan Teknik Elektro pada tahun 2012. Pada bulan Juni 2016 penulis mengikuti seminar dan ujian Tugas Akhir sebagai salah satu syarat untuk memperoleh gelar Sarjana Teknik Elektro dari Institut Teknologi Sepuluh Nopember Surabaya.